

Programmers Reference Manual

DDP-416

GENERAL PURPOSE COMPUTER

August 1967

Honeywell



COPYRIGHT 1967 by Honeywell Inc., Computer Control Division, Framingham, Massachusetts. Contents of this publication may not be reproduced in any form in whole or in part, without permission of the copyright owner. All rights reserved.

Printed in U.S.A.

CONTENTS

Page

SECTION I COMPUTER ORGANIZATION

Specifications	1-1
System Description	1-2
Word Formats	1-5
Data Formats	1-5
Memory Addressing	1-8
Direct Addressing	1-8
Indirect Addressing	1-9
Locations $(00001)_8$ to $(00017)_8$	1-9
Memory Reference Instruction Logic and Timing	1-10

SECTION II STANDARD INSTRUCTIONS

Instruction Repertoire	2-1
------------------------	-----

SECTION III INPUT/OUTPUT

Input/Output Control and Communication	3-1
Single-Word Transfer Mode	3-3
Standard Interrupt	3-4
Power Failure Interrupt (PFI)	3-6

SECTION IV MAIN FRAME OPTIONS

Memory Parity Option, Model 416-07	4-1
Instruction Complement	4-1
Memory Lockout Option, Model 416-08	4-1
Base Sector Relocation	4-2
Restricted Mode	4-4
Normal Mode	4-5
Protected Sector Selection	4-5
Real Time Clock Option, Model 416-12	4-6
Instruction Complement	4-6
Direct Multiplex Control, Model 416-20	4-7
DMC Sub-Channel	4-9
DMC Auto-Switch Option	4-10
Direct Memory Access Option, Model 416-21	4-10
Applications	4-10
Instruction Complement	4-11

CONTENTS (Cont)

	<u>Page</u>
Programming	4-12
DMA Auto-Switch	4-12
Priority Interrupt Option, Model 416-25/25-1	4-12
Priority Interrupt Control	4-13
Memory Increment Option, Model 416-26	4-14
Memory Access Priority Structure	4-15
Program Interrupts	4-15
Computer Breaks	4-16

SECTION V PERIPHERAL DEVICES

ASR-33/35 Teletype Units, Model 416-53/55	5-1
Keyboard and Carriage Features	5-1
Operating Modes, ASR-35 Unit	5-2
Reader Control (Effective in KT and T Modes Only)	5-3
On-Line Punch Control	5-3
Off-Line (Local) Punch Control	5-3
Operating Modes - ASR-33 Unit	5-4
Tape Reader	5-4
Tape Punch	5-4
Off-Line Operation	5-4
ASR-33/35 On-Line Operating Modes	5-5
Character Modes	5-5
Instructions	5-6
Standard Interrupt	5-7
Paper Tape Format and ASR Codes	5-8
High-Speed Paper-Tape Reader, Model 416-50	5-11
Reader Modes	5-11
Codes	5-11
Specifications	5-11
Commands	5-11
High-Speed Paper-Tape Punch, Model 416-52	5-15
Specifications	5-15
Loading Procedure	5-15
Commands	5-15

SECTION VI SAMPLE PROGRAMS

Key-In Loader for ASR-33/ASR-35	6-1
Fixed Point, Double Precision Add Subroutine	6-2

CONTENTS (Cont)

	<u>Page</u>
Fixed Point, Double Precision Subtract Subroutine	6-3
Fixed Point, Double Precision Multiply Subroutine	6-4
Fixed Point, Double Precision Divide Subroutine	6-6
Output on ASR-33	6-8
Paper Tape Read Subroutine	6-8
Output on High-Speed Paper Tape Punch	6-9

SECTION VII OPERATION

Control Console	7-1
Operating Procedures	7-1
Turn-On, Turn-Off	7-1
Initialize System	7-1
Read Single Memory Location	7-6
Read Consecutive Memory Locations	7-7
Single Memory Location Data Insertion and/or Modification	7-7
Consecutive Memory Location Data Insertion and/or Modification	7-8
Single-Step Program Execution	7-8
Read Mainframe Register	7-8
Display Operational Data	7-9
Run Program	7-9

APPENDICES

		<u>Page</u>
A	Numbering System and Two's Complement Arithmetic	A-1
B	ASCII Code	B-1
C	Summary of Standard Instructions (Listed in Alphabetical Order)	C-1
D	Peripheral Device Commands	D-1
E	Main Frame Option Commands	E-1
F	Dedicated Locations	F-1

ILLUSTRATIONS

		<u>Page</u>
1-1	DDP-416 Simplified Block Diagram	1-4
1-2	Data Word Format, Single Precision	1-5
1-3	Data Word Format, Double Precision	1-5
1-4	Memory Reference Instruction Format	1-6
1-5	Input/Output Instruction Format	1-6
1-6	Shift Instruction Format	1-6
1-7	Generic Instruction Format	1-7
1-8	Memory Sectors in 4096-Word DDP-416	1-8
1-9	Indirect Address Format	1-9
1-10	Fetch and Indirect Addressing, Logic Flow Diagram	1-11
4-1	Base Sector Relocation Relative to Indirect Addressing, Flow Diagram	4-3
5-1	ASR-33/35 Paper Tape Format	5-8
5-2	High-Speed Paper-Tape Reader, Model 416-50	5-12
5-3	High-Speed Paper-Tape Punch, Model 416-52	5-16
7-1	Control Console, Control Panel	7-2

TABLES

	<u>Page</u>
2-1 Glossary of Symbols	2-2
2-2 DDP-416 Instruction Repertoire	2-3
3-1 Input/Output Bus Lines	3-1
3-2 Standard Interrupt Mask Assignments	3-5
4-1 Memory Parity Instructions	4-1
4-2 Memory Lockout Instructions	4-4
4-3 Protected Memory Ranges	4-5
4-4 Real Time Clock Option Instruction Complement	4-7
4-5 DMC Start and Terminal Memory Address Locations	4-8
4-6 Direct Memory Access Instructions	4-11
4-7 Dedicated Locations for the Twelve Groups of Priority Interrupt Lines	4-13
4-8 Priority Interrupt Mask Assignments	4-14
4-9 DDP-416 Computer Access-to-Memory Priority Structure	4-15
5-1 ASR-33/35 Characters and Symbol Codes	5-9
7-1 Control Panel Controls and Indicators	7-3
7-2 Control Panel Data Bit Indicators, Displayed Operational Functions and Indicator Symbols	7-6



DDP-416 General Purpose Computer

INTRODUCTION

The DDP-416 is an integrated circuit 16-bit binary word general purpose digital computer with a 0.96- μ sec cycle time magnetic core memory. The DDP-416 has a fully parallel machine organization and multilevel indirect addressing. Memory sizes available are 4096, 8192, 12,288, and 16,384 words. Standard features include a flexible instruction repertoire of 30 commands, a powerful I/O bus structure, and standard Teleprinter keyboard and paper tape I/O unit. An extensive programming package, including a symbolic assembler, subroutine linking loader, math library, I/O library and diagnostic and utility routines, is provided with the basic DDP-416. Options include memory parity, memory lockout, priority interrupt, a direct multiplex control unit, direct memory access, a real-time clock, and a full line of peripheral equipment.

The 16-bit word of the DDP-416 allows a straightforward and efficient addressing scheme. Most internal operations can be performed in two cycle times (1.92 μ sec), or less including instruction access and execution time. A single word instruction can directly address any one of 1024 words. The 16-bit word is directly compatible with the ASCII 8-bit character code.

The DDP-416 is designed for real-time on-line data processing and control. Modular design and a flexible I/O structure and a standard repertoire of 30 instructions enable the DDP-416 to be tailored to control, communications and data acquisition applications.

Programming the DDP-416 computer is similar to programming other single-address binary computers using two's complement notation. Therefore, no major differences confront the programmer who is new to the DDP-416.

SECTION I
COMPUTER ORGANIZATION

SPECIFICATIONS

Type

Parallel binary

Addressing

Single address with
indirect addressing

Word Length

16 bits

Machine Code

Two's complement

Memory Type

Magnetic core

Memory Size

4,096, 8,192, 12,288, or 16,384

Memory Cycle Time

0.96 μ sec

Speed

Add: 1.92 μ sec

Subtract: 1.92 μ sec

Standard Peripheral Equipment

ASR-33 or 35 Teletype Unit providing the following capabilities:

- a. Read paper tape at 10 cps
- b. Punch paper tape at 10 cps
- c. Type at 10 cps
- d. Keyboard input
- e. Off-line paper-tape preparation, reproduction and listing

Optional Peripheral Equipment

300 cps photoelectric paper-tape reader

110 cps paper-tape punch

300 line-per-minute (120-character-per-line) high-speed printer

200 card-per-minute card reader

Magnetic tape units:

<u>Unit</u>	<u>Tape Speed (ips)</u>	<u>Density (bpi)</u>
Low speed	36	200, 556, 800
High speed	80	200, 556, 800

Standard Input/Output Lines

16-bit input bus

16-bit output bus

10-bit device address bus

External control and sense lines

Input/Output Modes

Three modes are available for data transfer between peripheral devices and the DDP-416.

- a. Single word transfer with or without interrupt
- b. Direct multiplex control (DMC) (optional)
- c. Direct memory access (DMA) (optional)

Interrupt

Single interrupt line standard. Up to 48 optional priority interrupts are available.

Power Failure Protection

Power failure interrupt standard. Core memory protected against loss of information on ac power failure.

SYSTEM DESCRIPTION

Figure 1-1, a block diagram of the computer, shows the data storage registers, the control unit of the central processor and the input/output controls. The random access memory, shown as a single block, is a magnetic core unit containing one or more memory modules of 4096 or 8192 16-bit words. Data from the memory is transferred to and from the DDP-416 registers through the M-register. The functional units of the central processor and the input/output controls are as follows:

A-Register (A): A 16-bit register used as the primary arithmetic and logic register of the computer.

Program Counter (P): A 16-bit register that contains the location of the next instruction to be executed.

Adder: Performs the basic arithmetic processes of addition and subtraction.

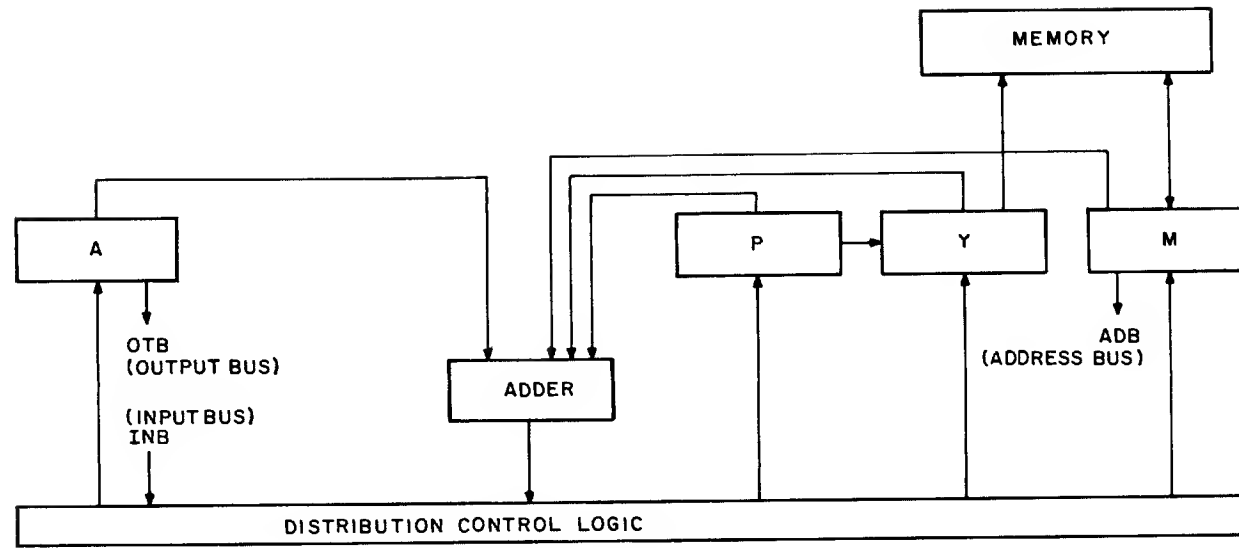
M-Register (M): A 16-bit register used to transfer information to and from the magnetic core memory.

Y-Register (Y): A 16-bit register used to store the address for the memory.

Output Bus (OTB): Sixteen lines that transmit data from the computer A-register to an I/O device.

Input Bus (INB): Sixteen lines that transmit data from an I/O device to the computer A-register.

Address Bus (ADB): Ten lines used in conjunction with I/O devices. Bits on lines 7 through 10 define the function to be performed by the I/O device. Bits on lines 11 through 16 designate the I/O device to be used.



A3936

Figure 1-1. DDP-416 Simplified Block Diagram

WORD FORMATS

Data Formats

Single Precision. -- The format for data words stored in the computer is shown in Figure 1-2.

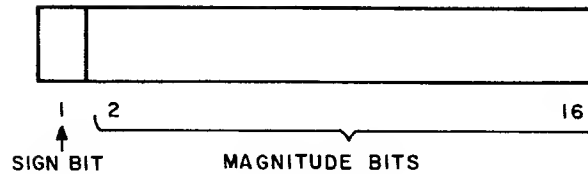


Figure 1-2. Data Word Format, Single Precision

Sixteen-bit data words are stored in two's complement form. The first bit of a data word may be considered the arithmetic sign and is zero for positive data.

Double Precision. -- When greater precision is required than that obtainable when using the single precision format, the double precision format is used (Figure 1-3). The sign position of the second (least significant) word is always zero. Thirty bits of magnitude are obtainable.

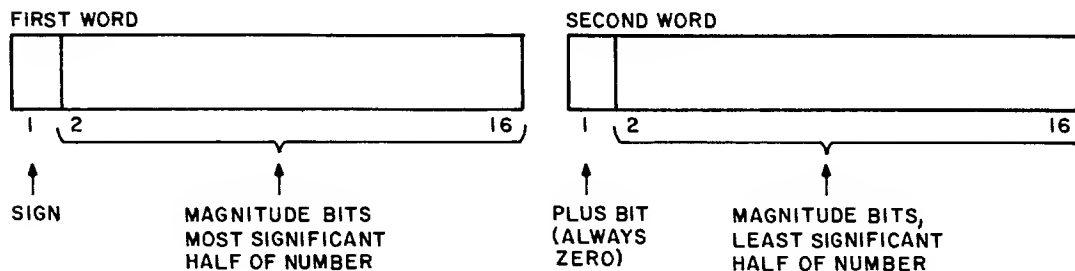


Figure 1-3. Data Word Format, Double Precision

Logical Data. -- Logical data, such as the condition of sixteen binary indicators, can be stored in a single data word. This type of data is generally not treated arithmetically by the program but logically by means of Boolean operators such as "AND" and "exclusive OR." In this case, bit 1 of a word does not represent the sign but the first of sixteen conditions.

Instruction Words. -- Instruction words are divided into four types: memory reference, input-output, shift, and generic.

The generic instruction word format is shown in Figure 1-7. All 16 bits are used to specify the instruction.

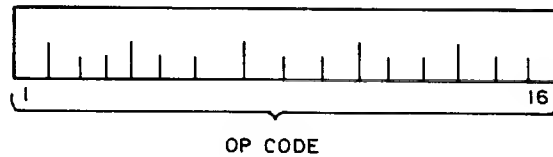


Figure 1-7. Generic Instruction Format

The op code expressed in binary, octal, and mnemonic for representative instructions of each of the four types, are listed in the following:

<u>Instruction</u>	<u>Type</u>	<u>Binary</u>	<u>Operation Code</u>	
			<u>Octal</u>	<u>Mnemonic</u>
Subtract	Memory Reference	x x 0 111 x xxx xxx xxx	07	SUB
Input to A	Input/Output	101 100 x xxx xxx xxx	54	INA
Arithmetic Left Shift	Shift	0 100 001 101 xxx xxx	0415	ALS
Clear A	Generic	1 100 000 000 100 000	140040	CRA

MEMORY ADDRESSING

Two techniques are used in the DDP-416 for memory addressing: direct addressing and indirect addressing.

Direct Addressing

The memory of the DDP-416 is considered to be divided into sectors of 512 words each. A 4096-word computer will have 8 sectors; an 8192-word computer, 16. Any word in a sector can be addressed with 9 bits ($2^9 = 512$). The address portion of a memory reference instruction (bits 8 to 16) can thus define a unique word in a sector. Addresses within sectors run from $(000)_8$ to $(777)_8$. The sector bit, bit 7 of the instruction, identifies the sector of the word addressed in accordance with the following rules:

Sector Bit = 0	The address is in sector 0 (octal address 00000 - 00777).
Sector Bit = 1	The address is in the same sector as the instruction being executed.

For example, assume an ADD 444 instruction is in address $(02100)_8$, or sector 2 word 100. If the sector bit in the instruction is 0, the instruction references word 444 in sector 0, or $(00444)_8$. If the sector bit is 1, then the instruction references word 444 in sector 2, or $(02444)_8$, because the instruction itself is in sector 2.

A single instruction can thus directly address 1024 words, half of which are in sector 0 and half of which are determined by the location of the instruction. Figure 1-8 represents the memory that can be directly addressed by an instruction in sector 2 and an instruction in sector 6.

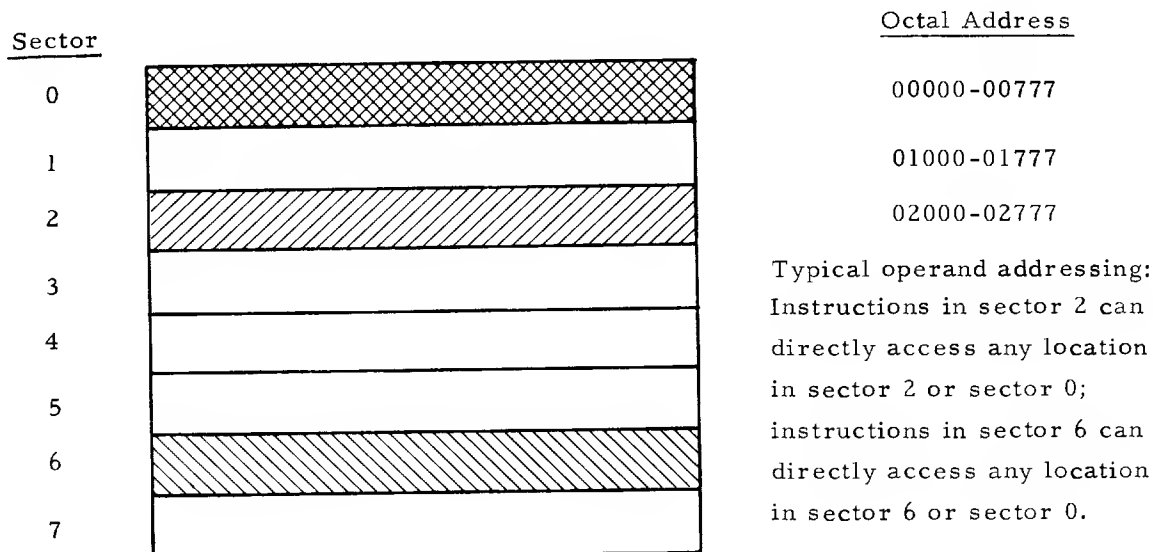


Figure 1-8. Memory Sectors in 4096-Word DDP-416

Indirect Addressing

If bit 1 of a memory reference instruction is set, indirect addressing takes place. When indirect addressing is specified, the effective address of the operand is assumed to be in the location specified by the address portion of the instruction and the sector bit. The format of the indirect address location is shown in Figure 1-9.

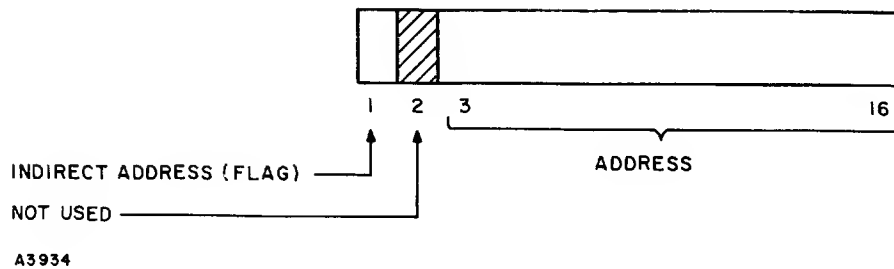


Figure 1-9. Indirect Address Format

To illustrate indirect addressing, consider that an add command in sector 2 is flagged for indirect addressing (this is specified in DAP-16 by placing an asterisk after the op code).

ADD* 444

Location 444 contains

$(06231)_8$

The effective address would then be $(06231)_8$, which is in sector 6. The content of location 06231 would be added to the A-register.

If the indirect bit within an indirect address location is set, a further level of indirect addressing takes place. This chaining of indirect addresses can continue indefinitely.

Locations $(00001)_8$ to $(00017)_8$

Memory locations $(00001)_8$ through $(00017)_8$ are protected in the standard machine against being written into under program control. Information may be read from these locations in the normal manner, however, all instructions which attempt to write in them will be aborted. The only way in which these locations may be loaded is through the use of the memory access feature of the console (see Section VII). The locations provide protected storage for the Key-In Loader utilized with the software system.

A Sample Key-In Loader is given in Section VI.

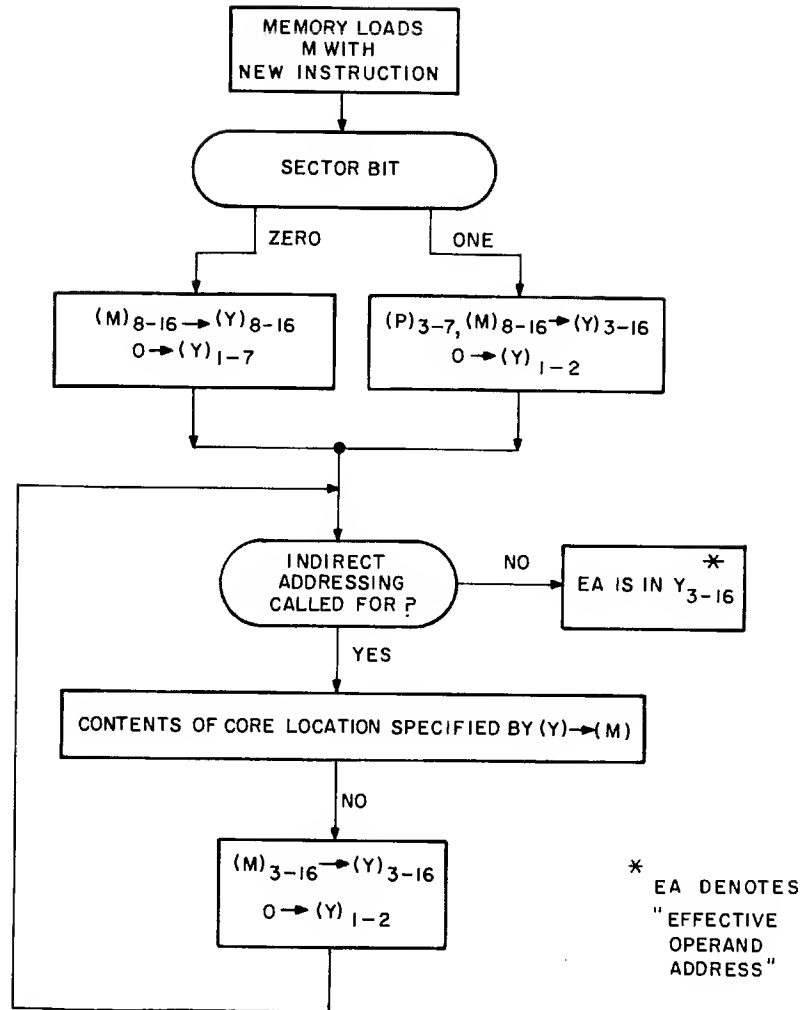
MEMORY REFERENCE INSTRUCTION LOGIC AND TIMING

Figure 1-10 is a logic flow diagram of the fetch and indirect addressing phases of an instruction. Initially, the P-register (program counter) contains the address of the instruction to be executed. The Y-register (memory address) also contains the same address. The instruction in the address specified by Y is then read out of memory into the M-register (memory information), and the operation code stored in the op-code register.

The sector bit is first examined. If the sector bit is set, the seven most significant bits of the program counter (the sector the instruction came from) and the 9 least significant bits of M (the address portion of the instruction) are transferred to Y. If the sector bit is ZERO, ZEROS are placed in the seven most significant bits of Y (thus addressing sector zero). If the indirect bit is not set, no indirect addressing is required and the contents of Y represent the effective operand address of the instruction. The computer then proceeds to the execution phase of the instruction.

If, when the indirect bit is examined, it is a ONE, indirect addressing is required. The contents of Y formed as a result of examination of the sector bit is then treated as the address of an indirect address word in memory rather than the effective operand address. The indirect address is then read out of memory (into the M-register) and its 14 least significant bits placed in the Y register.

If the indirect bit in the indirect address word is a ZERO, the contents of Y represent the effective operand address of the instruction. If the indirect bit is a ONE, Y represents the address of another indirect address word which is read out of memory and processed in the same manner as the first. There is no basic limit to the number of indirect words which can be called for before the generation of the effective operand address.



A3937

Figure 1-10. Fetch and Indirect Addressing,
Logic Flow Diagram

SECTION II STANDARD INSTRUCTIONS

INSTRUCTION REPERTOIRE

The instructions which comprise the standard DDP-416 Instruction Repertoire are described in detail in this section. Mnemonics and symbols used in the instruction descriptions are listed in Table 2-1. A thorough knowledge of the data presented in Table 2-1 is necessary to understand the instruction descriptions.

Table 2-2 lists all standard instructions. Each instruction is identified by its assigned three-letter mnemonic, type symbol, and octal Op-Code. Definitions, descriptions, and timing data for each instruction are also included in Table 2-2. Refer to Section I for instruction word formats.

The standard instructions in Table 2-2 are grouped into the following operational categories:

- Load and Store
- Arithmetic
- Logical
- Shift
- Input/Output
- Control

Table 2-1.
Glossary of Symbols

Symbols	Definition
EA	Effective operand address; the address from which the operand will be obtained. This is determined only after all selection of sectors and indirect addressing required have been performed.
n	Specified number of shifts to be performed
N	Two's complement of the number of shifts to be performed.
ADB	Address Bus
INB	Input Bus
OTB	Output Bus
A	A-Register (16-bits)
P	Program Counter (16-bits)
M	M-Register (16-bits)
→	Replaces
→	Is discarded
∧	Logical AND
∨	Logical OR
⊕	Exclusive OR
+	Algebraic Addition
()	Contents of a hardware register (e.g., (A) = contents of A-Register)
[]	Contents of core location specified by (e.g., [EA] = contents of core location specified by EA)
MR	Memory Reference Instruction
G	Generic Instruction
SH	Shift Instruction
IO	Input-Output Instruction

Table 2-2.
DDP-416 Instruction Repertoire

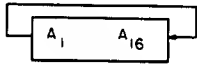
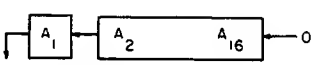
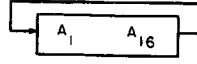
Mnemonic	Type	Op Code	Definition	Description	No. of Cycles	Time (μ sec)
Load and Store						
CRA	G	140040	Clear A	$0 \rightarrow (A)$	1	0.96
LDA	MR	02	Load A	$[EA] \rightarrow (A)$	2	1.92
STA	MR	04	Store A	$(A) \rightarrow [EA]$	2	1.92
Arithmetic						
ADD	MR	06	Add	$(A) + [EA] \rightarrow (A)$	2	1.92
SUB	MR	07	Subtract	$(A) - [EA] \rightarrow (A)$	2	1.92
Logical						
ANA	MR	03	AND to A	$(A) \wedge [EA] \rightarrow (A)$	2	1.92
ERA	MR	05	Exclusive OR to A	$(A) \vee [EA] \rightarrow (A)$	2	1.92
Shift						
ALR	SH	0416	Logical Left Rotate	 <p>The A register is shifted left, end-around (n) positions. A_1 is shifted out to A_{16}.</p>	$1 + n/2$	$0.96 + 0.48n$
ALS	SH	0415	Arithmetic Left Shift	 <p>The A register is shifted left (n) positions. After 16 or more shifts, the A register contains ZERO.</p>	$1 + n/2$	$0.96 + 0.48n$
ARR	SH	0406	Logical Right Rotate	 <p>The A register is shifted right, end-around (n) positions. Bits shifted out of A_{16} enter A_1.</p>	$1 + n/2$	$0.96 + 0.48n$
					$1 + n/2$	$0.96 + 0.48n$

Table 2-2. (Cont)
DDP-416 Instruction Repertoire

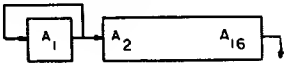
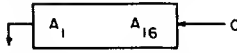
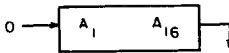
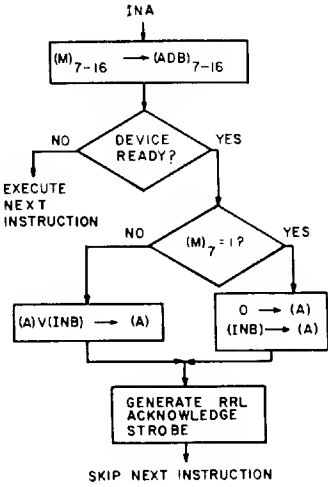
Mnemonic	Type	Op Code	Definition	Description	No. of Cycles	Time (μsec)
ARS	SH	0405	Arithmetic Right Shift	 <p>The A register is shifted right (n) positions. The sign bit (A₁) does not change; it is shifted into vacated positions of the register. Bits shifted out of A₁₆ are discarded. If 15 or more shifts are specified, all stages of the A register will be the same as the sign bit.</p>	1 + n/2	0.96 + 0.48n
LGL	SH	0414	Logical Left Shift	 <p>The A register is shifted left (n) positions. ZEROs fill in vacated bit positions. After 16 or more shifts, the A register contains ZERO.</p>	1 + n/2	0.96 + 0.48n
LGR	SH	0404	Logical Right Shift	 <p>The A register is shifted right (n) positions. ZEROs fill in vacated bit positions. After 16 or more shifts, the A register contains ZERO.</p>	1 + n/2	0.96 + 0.48n
Input-Output For I/O Discussion see Section III						
INA	IO	54 For INA Codes see Appendix D	Input to A		2	1.92

Table 2-2. (Cont)
DDP-416 Instruction Repertoire

Mnemonic	Type	Op Code	Definition	Description	No. of Cycles	Time (μ sec)
OCF	IO	14 For OCF codes see Appendix D	Output Control Pulse	<pre> graph TD A["(M)7-16 -> (ADB)7-16"] --> B[GENERATE OCF CONTROL PULSE] B --> C[EXECUTE NEXT INSTRUCTION] </pre>	2	1.92
OTA	IO	74 For OTA codes see Appendix D	Output from A	<pre> graph TD A[OTA] --> B["(M)7-16 -> (ADB)7-16"] B --> C{DEVICE READY?} C -- YES --> D["(A) -> (OTB)"] D --> E[GENERATE RRL OUTPUT AND ACKNOWLEDGE STROBE] E --> F[SKIP NEXT INSTRUCTION] C -- NO --> G[EXECUTE NEXT INSTRUCTION] </pre>	2	1.92
SMK	IO	74 For SMK codes see Appendix D	Set Mask (Special OTA)	<p>(A) \rightarrow (OTB)</p> <p>Generate SMK pulse to transfer output bus to external device mask flip-flops. This instruction does not skip.</p>	2	1.92
SKS	IO	34 For SKS codes see Appendix D	Skip if Ready Line Set	<pre> graph TD A["(M)7-16 -> (ADB)7-16"] --> B{READY?} B -- YES --> C[SKIP NEXT INSTRUCTION] B -- NO --> D[EXECUTE NEXT INSTRUCTION] </pre>	2	1.92

Table 2-2. (Cont)
DDP-416 Instruction Repertoire

Mnemonic	Type	Op Code	Definition	Description	No. of Cycles	Time (μ sec)
Control						
ENB	G	000401	Enable Program Interrupt	Set machine status to permit interrupt. The permit interrupt status will not take effect until the instruction immediately following ENB is completed. (PI indicator lights.)	1	0.96
HLT	G	000000	Halt	Sets machine to halt mode. No further instructions or interrupts will be serviced until the console START button is pressed, at which time normal execution resumes.	1.5	1.44
INH	G	001001	Inhibit Program Interrupt	Resets "permit interrupt status" to prohibit standard or priority interrupts. (PI indicator is extinguished.)	1	0.96
IRS	MR	12	Increment, replace and Skip	$[EA] + 1 \rightarrow [EA]$ If $[EA] + 1 = 0$, skip next instruction	3	2.88
JMP	MR	01	Unconditional Jump	$EA \rightarrow (P)$ Next instruction to be executed is at location EA.	1	0.96
JST	MR	10	Jump and Store Location	$(P_{3-16}) \rightarrow [EA_{3-16}]$ $[EA_{1,2}]$ not changed $EA_{3-16} + 1 \rightarrow (P_{3-16})$	3	2.88
NOP	G	101000	No Operation	Performs no operation. Computer proceeds to next instruction.	1	0.96
SKP	G	100000	Unconditional Skip	Skip next instruction.	1	0.96

Table 2-2. (Cont)
DDP-416 Instruction Repertoire

Mnemonic	Type	Op Code	Definition	Description	No. of Cycles	Time (μ sec)
SMI	G	101400	Skip if A Minus	If (A_1) = 1: skip next instruction	1	0.96
SNZ	G	101040	Skip if A Not Zero	If (A) \neq 0: skip next instruction	1	0.96
SPL	G	100400	Skip if A Plus	If (A_1) = 0: skip next instruction	1	0.96
SZE	G	100040	Skip if A Zero	If (A) = 0: skip next instruction	1	0.96

SECTION III INPUT/OUTPUT

INPUT/OUTPUT CONTROL AND COMMUNICATION

The basic communication link between the computer and peripheral (input/output) devices is an input/output bus. This bus contains 16 input lines, 16 output lines, 10 address lines and a group of control lines. As many as 20 peripheral devices may be attached to the bus. These devices then all communicate with the central processor by time sharing the bus. Since all computer standard I/O devices are individually buffered, and the bus is only used by a particular device while the computer is actually transferring information to or from the device, many devices can operate concurrently. The input/output bus lines are listed in Table 3-1.

Table 3-1.
Input/Output Bus Lines

Lines Available for Input/Output	Designation	Bit Capacity	Function
Output bus	OTB ₁₋₁₆	16	Transmit data from the computer to an I/O device
Input bus	INB ₁₋₁₆	16	Transmit data from an I/O device to the computer
Address bus	ADB ₇₋₁₀	4	Define the function to be performed by an I/O device
	ADB ₁₁₋₁₆	6	Define the I/O device selected
Device ready line	DRLIN	1	Transmit a signal to the computer indicating the status of the device addressed by the I/O command
Output control pulse	OCPLS	1	Transmit a pulse from the computer that defines the fact that an OCP command is being executed
Master clear	MSTCL	1	Transmit a master reset to devices
Parity error	PARCK	1	Transmit a signal to the computer indicating that a parity error has been detected in an I/O device
Program-interrupt line	PIL00	1	Transmit a signal to the computer indicating that a standard interrupt is requested

Table 3-1. (Cont)
Input/Output Bus Lines

Lines Available for Input/Output	Designation	Bit Capacity	Function
Set Program interrupt mask	SMK01	1	Transmit a pulse from the computer indicating that the OTB contains a new setting for the interrupt mask flip-flops
Clear mask	CMKXX	1	Transmit a pulse from the computer used to clear the device mask control flip-flops
Set Mask (general)	SMKXX	1	Transmit a pulse from the computer indicating that the OTB contains a new setting for option masks specified by ADB _{7-10; 14}
Reset ready line	RRL1N	1	Transmit a pulse from the computer which is used to strobe the output bus during an OTA instruction and to "reset ready" during the OTA and INA instructions.

The central processor is responsible at all times for determining what information is on the bus. Thus, the typical sequence of operation is for the computer to send out on the address bus lines a 6-bit device code that identifies the device with which the central processor is communicating and a 4-bit function code indicating which function the device is to perform. If the instruction is an input to A (INA), output from A (OTA), or sense status (SKS), the device next sends back to the central processor an indication as to its condition (ready, etc.). The central processor then performs the necessary functions (input, output, skip, etc.) on the basis of the reply.

The selection of a device, the testing for its status, and the actual input or output can often be performed with a single instruction. Once each device has been set up in its proper operating mode and been started, the only instructions necessary to perform data transfers are INA or OTA instructions.

Three basic modes of input/output are available with the computer. The standard mode is single-word input/output transfer, with or without interrupt. The second mode is the optional DMC (direct multiplex control) which permits input/output to and from memory without program intervention. The third mode is the optional DMA (direct memory access), which can be used with special devices to achieve very high transfer rates.

Single-Word Transfer Mode

The single-word transfer under program control is the basic input/output mode of the computer. In this mode, full words or character can be read from external devices into the A-register by utilizing INA instructions, and words or characters can be transferred from the A-register to an output device by using OTA instructions. During an input operation in the single-word transfer mode, the programmer has the option of clearing or not clearing the A-register before each input. If characters are being read into the computer, this allows the programmer to pack the characters into words in the A-register as part of a basic input routine. The ability to test and skip on the ready status of an I/O device also is included in the basic input and output instructions to make the computer extremely flexible in real-time applications. Thus, the computer is not required to hold in an input or output instruction waiting for a ready signal. This permits maximum utilization of the central processor. It also makes it convenient to handle multiple input/output devices all running simultaneously under program control. Because of the high internal speed of the computer, quite high data transfer rates can be accommodated in the single-word transfer mode. This mode is also convenient for slower devices such as paper-tape equipment and card equipment.

The instructions which are used to operate in the single-word transfer mode are as follows.

- a. Input to A (INA)
- b. Output from A (OTA)
- c. Sense status (SKS)
- d. Output control pulse (OCP)

On each of these instructions bits 11 through 16 identify the I/O device selected, and bits 7 through 10 define the function to be performed. With the exception of bit 7 in the INA command, these bits are completely ignored by the central processor. Their only function is to serve as a command to the peripheral device.

INA Instruction. -- The INA instruction is used to input data from a device into the A-register. All 16 bits of the data are ORed into the A-register by the instruction; however, data is not necessarily placed on all lines by every device. Thus, a character input device may place data only on the eight least significant bits of the input bus leaving the other bits as ZEROS. Since the content of the input bus is always logically ORed with the A-register, the effect is as though only eight bits had been transferred from the device to the A-register. The function code portion of the INA instruction is typically used by the device to determine the mode of input (for example, binary or ASCII).

The INA instruction sends out its device and function code on the I/O bus. It then looks for a ready signal on the DRLIN (device ready line). If a ready signal is received within a predetermined time interval, the content of the INB (input bus) is logically ORed with the contents of the A-register and the next instruction is skipped. A reset-ready signal is also sent out on the RRLIN (reset ready line) to tell the device that the data has been

accepted by the computer. If bit 7 is set in the instruction, the A-register is cleared before the INB is ORed with the A-register. If a ready signal is not received, no input is performed and the next instruction is not skipped.

OTA Instruction. -- The OTA instruction is utilized to send data from the A-register to an output device. All 16 bits of the A-register are sent out on the I/O bus; however, not all may be accepted by a particular device. Thus, a character device might receive only the eight least significant bits of the data. The function code portion of the instruction is typically used by the device to determine the mode of output (for example, binary or ASCII).

This instruction sends out its device and function code and the contents of the A-register on the I/O bus. It then looks for a ready signal on the DRLIN (device ready line). If a ready signal is received within a predetermined time interval, an output pulse is sent out on the RRLIN line indicating to the device that it may take data off the OTB (output bus). The next instruction is then skipped. If a ready signal is not received, no output function is performed, and the next instruction is not skipped.

OCP Instruction. -- The OCP instruction is used to set up the operating mode of a device, to start the device, etc. This instruction sends out its device and function code on the I/O bus. It also sends an output control pulse on the OCPLS line after the device has had time to receive and decode the address and function bits. The function bits in this instruction are used to determine the particular function that the OCPLS pulse is required to perform. The DRLIN line is not examined during this instruction, and the next instruction is never skipped.

SKS Instruction. -- The SKS instruction is used to test different conditions in the device. Thus, it might test for "power on," "tape moving," "device busy," "device ready," etc. It is also used to supplement the device-ready test included in the INA and OTA instructions. The function bits are used to determine the particular condition to be tested.

This instruction sends out its device and function code on the I/O bus. It then looks for a status signal on DRLIN. If an affirmative status signal is received within the prescribed time interval, the next instruction is skipped. If an affirmative status signal is not received, the next instruction is not skipped.

Standard Interrupt

The basic interrupt system consists of a single interrupt line. All standard I/O devices are connected to this line. A total of 16 interrupt sources can be connected on this line. Each source also has an interrupt mask bit which can inhibit an interrupt signal from being gated onto the interrupt line. The mask can be set and reset by an SMK '0020 instruction, which transfers the contents of the A-register via the OTB to the mask bits of standard devices as listed in Table 3-2. Thus, the program has the ability to selectively inhibit interrupt sources. This selective inhibiting of interrupt sources permits a multilevel

priority interrupt system to be programmed in which an interrupt subroutine can be interrupted in turn by a program of even higher priority. Furthermore, because all interrupt sources connect with the computer via the I/O bus, the logic associated with all the interrupt sources does not have to be centralized in a priority interrupt unit; it can be located wherever it is most convenient to place it. In particular, I/O devices can be handled on a priority interrupt basis by merely adding the necessary logic to the device control unit.

When the interrupt line is activated by an external source, the computer inhibits all further interrupts and generates a jump and store location instruction (JST) indirectly through location $(00063)_8$. If more than one interrupt source is connected to the interrupt line, the program proceeds to an interrupt service routine which tests the sources one by one with sense status commands (SKS). When the routine finds the source which caused the interrupt, it jumps to the appropriate subroutine. The program then sets up a new status for the interrupt mask bits for all of the interrupt sources. The new status determines the sources that have a higher priority than the one which actually interrupted. The program then enables interrupt and proceeds.

The signals in the I/O bus which are used for interrupt are as follows.

- a. PIL00 - This ORs together interrupt request signals from all standard interrupt sources and sends them to the CPU.
- b. DRLIN - This line is used by the SKS instruction to test each individual interrupt source in order to check whether it is requesting an interrupt. The device address is sent out which selects the device, and a particular function code is sent out which places the status of the priority interrupt request logic on DRLIN.
- c. SMK01 - This line from the CPU is used in place of a device address and a function code to indicate that a new status for the interrupt mask bits in the system is on the OTB.

Table 3-2.
Standard Interrupt Mask Assignments

OTB Bit No.	Device	OTB Bit No.	Device
1	Mag Tape Control Unit No. 1	9	Paper Tape Reader
2	Mag Tape Control Unit No. 2	10	Paper Tape Punch
3	(Unassigned)	11	ASR-33/35
4	(Unassigned)	12	Card Reader
5	I/O Channel No. 1	13	(Unassigned)
6	I/O Channel No. 2	14	Line Printer
7	I/O Channel No. 3	15	Memory Parity
8	(Unassigned)	16	Real Time Clock

Power Failure Interrupt (PFI)

The basic computer contains a PFI circuit which acts as a memory protection feature. If the primary computer ac input power fails or is turned off at the control console while the computer is in the "RUN" mode, the PFI circuit either halts the computer or forces an interrupt to a pre-assigned memory location. The operation performed by the PFI on the detection of a power failure is dependent on the position of a console PFI/PFH control switch.

If the control switch is in the PFI position, the detection of a power failure will cause the PFI to initiate an interrupt during which the computer is forced to perform an indirect JST to memory location $(00060)_8$. The PFI interrupt will occur at least one millisecond before the dc power drops below the guaranteed operating limits of the circuits.

If the control switch is in the PFH position, the detection of a power failure causes the PFI to place the computer in a halt state. No information in memory will be altered when power fails.

SECTION IV MAIN FRAME OPTIONS

MEMORY PARITY OPTION, MODEL 416-07

The memory parity option enables generation of parity on all memory write cycles and checking of parity on all memory read cycles. An exception exists in that parity is not checked during a console memory read operation. The memory parity error flip-flop in the computer is set when a memory parity error occurs and can be tested and reset under program control. It can also be displayed on the computer control panel (refer to Section VII). The MSTR CLEAR pushbutton switch on the control panel resets the parity error flip-flop. When the parity error flip-flop is set, an interrupt is generated on the standard interrupt line. This interrupt can be masked on or off by the parity error mask bit (bit 15).

Instruction Complement

The instructions added when this option is included in a system are listed in Table 4-1.

Table 4-1
Memory Parity Instructions

Mnemonic	Type	Instruction Word	Definition	Description	No. of Cycles	Time (μsec)
RMP	G	000021	Reset Memory Parity Error	Resets memory parity error flip-flop	1	0.96
SPS	G	101200	Skip on Memory Parity Error	Skips next instruction if parity error flip-flop is set	1	0.96
SPN	G	100200	Skip on No Memory Parity Error	Skips next instruction if parity error flip-flop is reset	1	0.96
SMK '0020	I/O	170020	Set Mask	$(A)_{15} \rightarrow$ Parity Interrupt Mask 1. $(A_{15}) = 1$, enable interrupt 2. $(A_{15}) = 0$, inhibit interrupt	2	1.92

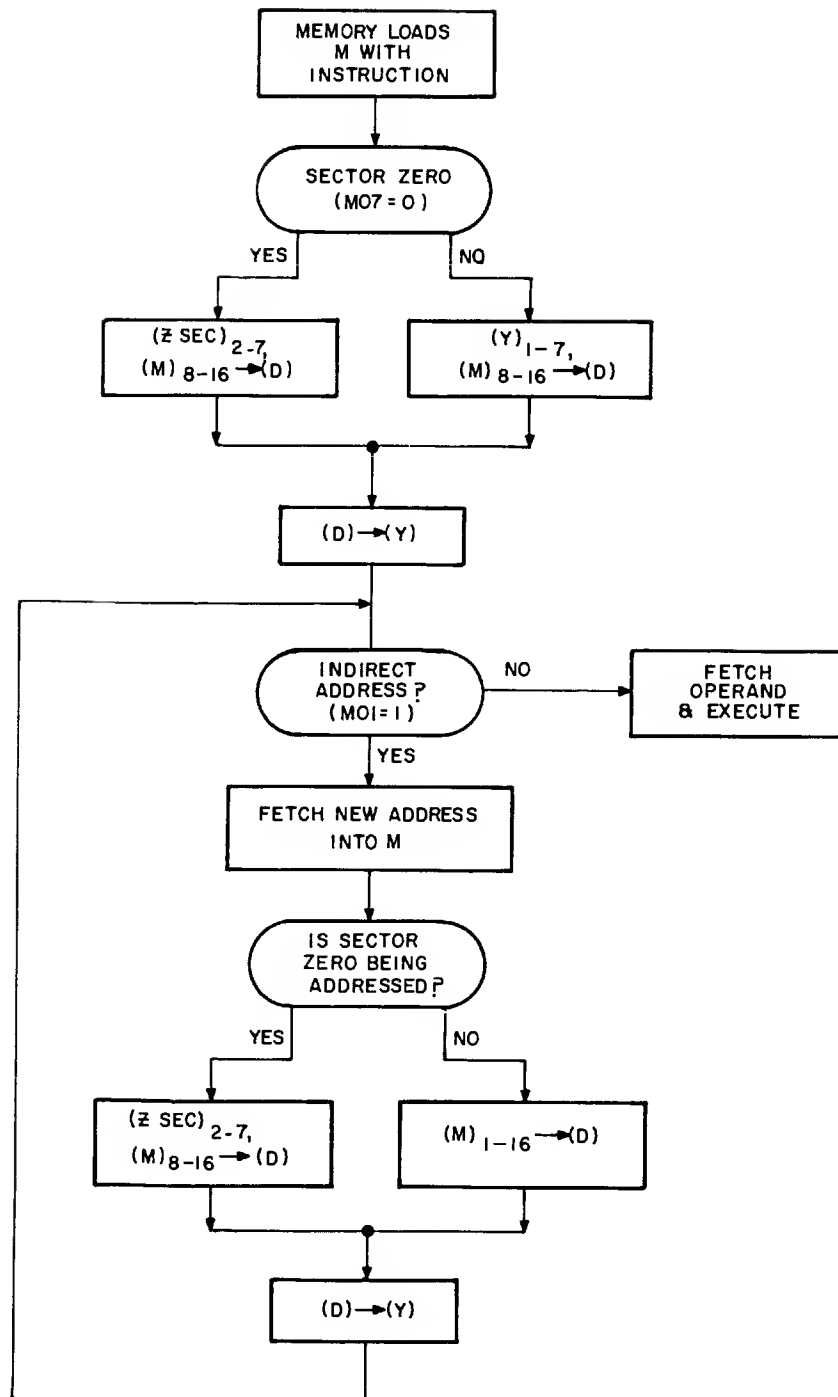
MEMORY LOCKOUT OPTION, MODEL 416-08

The memory lockout option facilitates the time-shared execution of various programs. The option provides base sector relocation to facilitate desectorization of more than one program. It also equips the CPU with a mode of operation called the "restricted mode" which enables unverified programs to be time-shared with other programs.

Base Sector Relocation

The memory lockout option provides for the relocation of the base sector insofar as the latter term applies to address information. The option includes a 6-bit base sector relocation register (ZSEC) used to identify the physical sector currently assigned as the base sector. When the sector bit, bit 7 of the instruction word, is a one, the address (bits 8-16) is in the same sector as the instruction being performed. This represents no change from the basic machine. When the sector bit is a zero, the memory lockout option forces the address to be in the sector specified by the base sector relocation register. Figure 4-1 contains a flow chart that shows when base sector relocation takes place relative to indirect addressing.

Base sector relocation does not affect memory references caused by breaks or program interrupts. The base sector relocation register can be changed by an SMK '1320 instruction (see Table 4-2). Any program interrupt, as well as MSTR CLEAR, clears this register.



A3938

Figure 4-1. Base Sector Relocation Relative to Indirect Addressing, Flow Diagram

Table 4-2
Memory Lockout Instructions

Mnemonic	Type	Instruction Word	Definition	Description	No. of Cycles	Time (μ sec)
ERM	G	001401	Enter Re- stricted Mode	Enables program inter- rupt and puts computer in restricted mode op- eration. Restricted operation continues un- til any program inter- rupt occurs. Does not take effect until after the next instruction is completed.	1	0.96
SMK '1320	I/O	171320	Set Relocation Register	(A) ₂₋₇ \rightarrow (ZSEC) ₂₋₇ Defines physical loca- tion of all address references to base sector until another SMK '1320 is executed or MSTR CLEAR is activated, or any interrupt occurs.	2	1.92
SMK '1420	I/O	171420	Set Lockout Mask 1	(A) ₁₋₁₆ \rightarrow (LMR) ₁₋₁₆	2	1.92
SMK '1520	I/O	171520	Set Lockout Mask 2	(A) ₁₋₁₆ \rightarrow (LMR) 17-32	2	1.92

Restricted Mode

There are two modes of operation associated with the memory lockout option; they are restricted and normal modes. The restricted mode has the following properties:

- a. Instructions which normally write into memory locations can be "locked out" of protected memory sectors. These instructions are: STA, IRS and JST.
- b. Certain instructions are considered illegal and cannot be performed. They are: OCP; SKS, OTA, INA, SMK, HLT, and INH.
- c. Indirect addressing is limited to eight levels.

Any instruction attempting to violate the above restrictions is aborted and causes a program interrupt to location (00062)₈. An aborted STA, SMK, OCP, HLT, or INH is executed as an NOP. SKS, OTA, and INA cause an interrupt but can skip as usual. A JST instruction does not store P but does jump to the effective address plus one. An IRS instruction does not increment memory but may skip the next instruction.

The program interrupt to location $(00062)_8$, generated by an aborted instruction, cannot be masked off, nor can it be tested with an SKS.

The restricted mode is entered by executing an ERM instruction. Visual indication of restricted mode operation is given by selecting the OP Register and observing the bit 11 indicator/pushbutton (refer to Table 7-2). Operation in the restricted mode is continuous until any program interrupt occurs. The MSTR CLEAR pushbutton places the machine in the normal mode.

The DMA, DMC, Real Time Clock and Memory Increment options are unaffected by the restricted mode since they are treated as agents of normal mode programs. This means that they can write in any memory location, even when they are sharing time with a program executed in the restricted mode.

Normal Mode

Normal mode operation is free of any restrictions and a program can execute any instruction in its repertoire.

Protected Sector Selection

Selection of those memory sectors which are to be protected is controlled by a lockout mask register (LMR). It is a 16-bit register (expandable to 32 bits) in which each bit is associated with one 512-word memory sector. A bit is zero if the corresponding sector is protected. The register is changed by an SMK instruction and cleared with the MSTR CLEAR pushbutton switch. Table 4-3 shows the specific memory ranges protected by SMK '1420 and SMK '1520.

Table 4-3
Protected Memory Ranges

A-Register Bit	SMK '1420	SMK '1520
1	00000-00777	20000-20777
2	01000-01777	21000-21777
3	02000-02777	22000-22777
4	03000-03777	23000-23777
5	04000-04777	24000-24777
6	05000-05777	25000-25777
7	06000-06777	26000-26777
8	07000-07777	27000-27777
9	10000-10777	30000-30777
10	11000-11777	31000-31777
11	12000-12777	32000-32777
12	13000-13777	33000-33777

Table 4-3 (Cont)
Protected Memory Ranges

A-Register Bit	SMK '1420	SMK '1520
13	14000-14777	34000-34777
14	15000-15777	35000-35777
15	16000-16777	36000-36777
16	17000-17777	37000-37777

NOTE

Locations 00001-00017 are always protected (in the basic main-frame) against all programs, restricted or normal. However, no MLO violation interrupt occurs if an attempt is made to write in these locations unless sector zero is protected and the machine is in the restricted mode.

REAL TIME CLOCK OPTION, MODEL 416-12

The Real Time Clock Option (RTC) permits the programmer to keep track of real time by automatically incrementing memory location $(00061)_8$. The frequency and stability of the incrementation is the same as the primary mainframe power source (50 or 60 cps \pm 2 cps). With a 60 cps power source, the RTC will increment location $(00061)_8$ every 16.67 ms. Incrementing can be enabled or disabled with an OCP '0020 or OCP '0220 instruction, respectively.

When memory location $(00061)_8$ overflows from $(17777)_8$ to $(000000)_8$ the RTC causes a program interrupt via the standard interrupt line. This program interrupt can be inhibited or enabled with an SMK '0020 instruction (refer to Standard Interrupt Description in Section III). OTB 16 (A register bit 16) is used to control the RTC interrupt. The interrupt can be tested by an SKS '0020 instruction and reset by an OCP '0220 or OCP '0020 instruction. If the RTC tries to interrupt when interrupt is masked off, it will wait until interrupt is enabled (by a proper SMK '0020 instruction) and then will cause an interrupt. Overflow from $(17777)_8$ to $(000000)_8$ does not inhibit incrementing.

Instruction Complement

The addition of the RTC option to a system adds three instructions to the basic system complement. The instructions which are added are described in Table 4-4.

Table 4-4.
Real Time Clock Option Instruction Complement

Mnemonic	Type	Instr Word	Definition	Description	No. of Cycles	Time (μsec)
OCP '0220	IO	030220	Reset Program Interrupt Request and Stop Clock	This instruction inhibits the RTC and resets the program interrupt request. One more real time clock break may occur immediately following this instruction if the increment request occurred during this instruction.	2	1.92
OCP '0020	IO	030020	Reset Program Interrupt Request and Run Clock	This instruction enables the RTC and resets the program interrupt request. The first increment request will occur within 0 to 16.7 ms following this instruction.	2	1.92
SKS '0020	IO	070020	Skip if RTC Not Interrupting	If the RTC is not requesting a program interrupt, the computer will skip the next instruction.	2	1.92

DIRECT MULTIPLEX CONTROL, MODEL 416-20

The Direct Multiplex Control (DMC) option permits data transfer between peripheral devices and the computer memory concurrently with computation.

When a device has data to input, or is ready to accept data, it uses the DMC control lines to request service. Devices request service from the DMC on lines called DIL. DIL line 1 has highest priority, line 16 has lowest. The priority network will allow the highest priority line which has its DIL set to be serviced by the next DMC cycle.

When a DMC cycle is required, the DMC will send a break request to the CPU. When the CPU has completed the current instruction, a DMC cycle will be executed. During this cycle the appropriate transfer between the device and the memory will take place, using the standard I/O bus.

This process is repeated each time the I/O device indicates that it is ready until the required number of words has been transferred. When the required number of words have been transferred, the DMC sends an End of Range (ERL) signal to the device. The device may use this signal to generate a program interrupt.

Up to 16 channels may be controlled by the DMC. Each channel requires a starting and ending address for the block transfer. These addresses (a pair per channel) are

stored in dedicated memory locations (listed in Table 4-5).

Bit 1 of the starting address is used to specify input or output mode. A one in bit 1 sets the DMC in the input mode. A zero in bit 1 sets the DMC in the output mode. The remaining fifteen bits specify the starting address of the data block. In input mode, data from the device will be stored beginning at this address. In output mode, data beginning at this address will be sent to the device. The high order bit of the final address is not interpreted. The remaining 15 bits specify the address into or out of which the final transfer will take place.

Table 4-5.
DMC Start and Terminal Memory Address Locations

Channel Number	Starting Address	Ending Address
1	00020	00021
2	00022	00023
3	00024	00025
4	00026	00027
5	00030	00031
6	00032	00033
7	00034	00035
8	00036	00037
9	00040	00041
10	00042	00043
11	00044	00045
12	00046	00047
13	00050	00051
14	00052	00053
15	00054	00055
16	00056	00057

The DMC can effect a transfer following any instruction, provided a DMC request from a device is transmitted to the DMC 0.6 μ sec before the end of that instruction. If a request occurs less than 0.6 μ sec before the end of an instruction, the DMC cycle may not occur until after the next instruction.

The data transfer is completed 1.74 μ sec into the DMC cycle for an input, 3.0 μ sec for an output. Thus, the longest waiting time, from the time a request occurs to the time the data transfer is completed is:

$$T_{wc} = T_{li} + 3.84M + 1.2N + \begin{matrix} 2.34 & \text{(input)} \\ 3.60 & \text{(output)} \end{matrix}$$

where

T_{wc} = worst-case waiting time (μsec) from request to completion of data transfer.

T_{li} = execution time of longest* instruction (μsec).

*The longest useful instruction in the CPU repertoire is executed in 8.64 μsec . (Shifts of more than 16 places and memory reference instructions with more than six levels of indirect addressing are not considered "useful" in this context.) Lower values of T_{li} may be used to facilitate input-output buffer design, provided appropriate programming constraints are adopted.

M = number of higher priority DMC requests which may occur during T_{wc} .

N = number of DMA requests which may occur during T_{wc} .

Each DMC cycle requires four memory cycles, or 3.84 μsec , during which computation is suspended. At 0.6 μsec before the end of a DMC cycle the device request lines are inspected. If a device is requesting at this time, another DMC cycle will immediately follow the first. DMC cycles will continue as long as requests are waiting. During this time the CPU cannot resume control.

The maximum transfer rate of a single DMC channel is one word every four cycles or 260 KC. This rate can be attained if this channel is the only channel being used. If the DMC is operating at 260 KC, no computation can take place. In order to operate between 200 KC - 260 KC, T_{li} must be 0.96 μsec .

The DMC may operate in the same system as a Direct Memory Access option with a restriction on the frequency of simultaneous operation as follows:

$$\sum_{i=1}^4 F_i + 4 \sum_{j=1}^{16} F_j < 1 \text{ mc/sec}$$

where

F_i = data transfer frequency of the devices which will operate simultaneously on Direct Memory Access Option; and

F_j = data transfer frequency of the devices that will operate simultaneously on DMC.

DMC Sub-Channel

A device is connected to the DMC control unit through a DMC sub-channel. The DMC sub-channel, available as an option on a number of standard I/O devices, contains the necessary logic to permit the device to operate in the DMC mode.

DMC Auto-Switch Option

The DMC Auto-Switch option provides automatic switching between two DMC sub-channels to permit the continuous transfer of data at high speed. To use the Auto-Switch option, one DMC sub-channel is set up as described above and the data transfer is started. While data is being transferred by the first DMC sub-channel, the second DMC sub-channel is set up. When the data transfer specified for the first sub-channel is complete, the Auto-Switch option automatically switches to the second DMC sub-channel and data transfer continues without interruption. An end-of-transmission interrupt occurs on the standard interrupt line to indicate that the switch has been made. The first DMC sub-channel must again be set up. When the data transfer specified for the second sub-channel is complete, the Auto-Switch option automatically switches back to the first sub-channel and interrupts. Switching is accomplished within one DMC cycle. This process is repeated continuously until the device is stopped or taken out of the DMC mode. Indicators associated with the device transferring data may be interrogated by the SKS instruction to determine which channel is active at any time and to determine which channel caused an interrupt.

DIRECT MEMORY ACCESS OPTION, MODEL 416-21

The direct memory access (DMA) option provides the central processor unit (CPU) with high speed input/output data transfer paths for addressing up to 16K of memory. The transfer rate approaches one word every 0.96 μ sec.

Applications

The DMA has the highest priority of all system options relative to memory access. The DMA is capable of interrupting between machine cycles such that any DMA interrupt request occurring during any cycle has access to memory at the end of that cycle. Note that the DMA is given access to memory without regard to whether or not the cycle just ended represents the completion of an instruction.

The DMA can effect a transfer following a memory cycle providing the request occurs 0.57 μ sec before the end of the cycle. However, requests arriving any later are serviced after the next memory cycle. The longest time between a request and the completion of the corresponding data transfer is 1.89 μ sec for input transfers and 2.64 μ sec for output transfers.

With few exceptions, all computation is momentarily suspended while a DMA cycle is in progress. The exceptions refer to the iterative instructions (e.g., LGL, etc.). These instructions comprise the shift/rotate group. The execution of these instructions continues simultaneously with the DMA transfer cycle.

A DMA can have from one to four channels. The channels are arranged in a priority network with Channel 1 having the highest priority and Channel 4 having the lowest priority.

Each channel has a 16-bit address counter which stores the starting address and a 16 bit range counter which stores the two's complement of the block size. The most significant bit (bit 1) of the starting address is used to specify input or output mode. A ONE in bit 1 sets the DMA in the input mode. The remaining 15 bits specify the memory address from which the first transfer will occur. The range and address counters are incremented each time a data transfer occurs. Range counter overflow signifies the completion of a block transfer. This is accomplished by the generation of an end-of-range signal which is sent to each device and can be used to cause a program interrupt. The contents of the range counters can be read into the computer to determine whether an external stop signal has terminated the DMA operation before the specified number of words are transferred.

Instruction Complement

A listing of the instructions required for use with the DMA option is presented in Table 4-6.

Table 4-6.
Direct Memory Access Instructions

Mnemonic	Type	Instruction Code	Definition	Description	No. of Cycles	Time (μsec)
SMK '0124	I/O	170124	Load Address Counter Channel 1	$(A)_{1-16} \rightarrow (AC1)_{1-16}$ $O \rightarrow (RC1)_{1-16}$	2	1.92
SMK '0224	I/O	170224	Load Address Counter Channel 2	$(A)_{1-16} \rightarrow (AC2)_{1-16}$ $O \rightarrow (RC2)_{1-16}$	2	1.92
SMK '0324	I/O	170324	Load Address Counter Channel 3	$(A)_{1-16} \rightarrow (AC3)_{1-16}$ $O \rightarrow (RC3)_{1-16}$	2	1.92
SMK '0424	I/O	170424	Load Address Counter Channel 4	$(A)_{1-16} \rightarrow (AC4)_{1-16}$ $O \rightarrow (RC4)_{1-16}$	2	1.92
SMK '1124	I/O	171124	Load Range Counter Channel 1	$(A)_{2-16} \vee (RC1)_{2-16}$ $\neg(RC1)_{2-16}$	2	1.92
SMK '1224	I/O	171224	Load Range Counter Channel 2	$(A)_{2-16} \vee (RC2)_{2-16}$ $\neg(RC2)_{2-16}$	2	1.92

Table 4-6. (Cont)
Direct Memory Access Instructions

Mnemonic	Type	Instruction Code	Definition	Description	Number of Cycles	Time (μsec)
SMK '1324	I/O	171324	Load Range Counter Channel 3	$(A)_{2-16} \vee (RC3)_{2-16}$ $\rightarrow (RC3)_{2-16}$	2	1.92
SMK '1424	I/O	171424	Load Range Counter Channel 4	$(A)_{2-16} \vee (RC4)_{2-16}$ $\rightarrow (RC4)_{2-16}$	2	1.92
INA '1124	I/O	131124	Read Range Counter Channel 1	If end-of-range, INA = NOP; otherwise, $1 \rightarrow (A)_1$ $(RC1)_{2-16} \rightarrow (A)_{2-16}$	2	1.92
INA '1224	I/O	131124	Read Range Counter Channel 2	If end-of-range, INA = NOP; otherwise, $1 \rightarrow (A)_1$ $(RC2)_{2-16} \rightarrow (A)_{2-16}$	2	1.92
INA '1324	I/O	131124	Read Range Counter Channel 3	If end-of-range, INA = NOP; otherwise, $1 \rightarrow (A)_1$ $(RC3)_{2-16} \rightarrow (A)_{2-16}$	2	1.92
INA '1424	I/O	131124	Read Range Counter Channel 4	If end-of-range, INA = NOP; otherwise, $1 \rightarrow (A)_1$ $(RC4)_{2-16} \rightarrow (A)_{2-16}$	2	1.92

Programming

The programming necessary for operating a device is:

Load Address Counter for Specific Channel (this will also clear the range register).

Load Range Register with two's complement of number of words to be transferred.

Activate Device.

DMA Auto-Switch

DMA Auto-Switch Option is available. It functions in a manner analogous to that previously described for the DMC Auto-Switch Option.

PRIORITY INTERRUPT OPTION, MODEL 416-25/25-1

A multilevel priority interrupt system is available as an option. This option eliminates the need for an interrupt service routine to determine which one of the available interrupt lines caused an interrupt. A unique memory location is dedicated to each interrupt

line. These locations are utilized in the same manner as the standard interrupt location is utilized in the standard interrupt system. When an interrupt occurs, the computer generates an indirect jump and store location instruction (JST) referencing the memory location dedicated to the source of the interrupt. Execution time of the computer-generated JST instruction is three cycles unless bit 1 of the dedicated location is a ONE. A ONE in this bit location indicates further indirect addressing; an additional cycle is required for each additional level of indirect addressing. Included in the option is a mask register which permits individual interrupt lines to be enabled and disabled under program control. This permits the relative priority of the interrupt lines to be established by the programmer.

The interrupt option is provided in groups of four interrupt lines. Up to 12 groups or a total of 48 interrupt lines can be handled by the system. The interrupt lines are consecutively numbered, and have decreasing priority with increasing number. The standard interrupt line is designated line 0 and retains its standard location $(63)_8$. The dedicated locations for the optional interrupt lines are shown in Table 4-7.

Priority Interrupt Control

Program interrupts requested by Priority Interrupt lines are individually controlled by mask bits associated with each group of interrupt lines. In addition, all Priority Interrupt lines are controlled by the INH and ENB instructions. Priority interrupt is inhibited until an ENB instruction has been executed. Following the execution of an ENB instruction, an interrupt will be accepted on any interrupt line having its associated mask bit set (one). Interrupt remains enabled until an INH instruction is executed or an interrupt occurs on any enabled line (forced INH). Following an interrupt or the execution of an INH instruction, interrupts will be inhibited until an ENB instruction is executed.

Table 4-7
Dedicated Locations for the Twelve Groups
of Priority Interrupt Lines

Priority Interrupt Group	Dedicated Locations (Octal Codes)
1	00064 - 00067
2	00070 - 00073
3	00074 - 00077
4	00100 - 00103
5	00104 - 00107
6	00110 - 00113
7	00114 - 00117
8	00120 - 00123
9	00124 - 00127
10	00130 - 00133
11	00134 - 00137
12	00140 - 00143

The mask bits associated with each group of interrupt lines are controlled by SMK '0X20 instructions. These instructions set the appropriate bit in the mask register if the corresponding bit in the A-register is a ONE and reset the mask register bit if the corresponding A-register bit is a ZERO. Table 4-8 shows the mask assignments for the optional interrupt lines and the SMK instructions that service them.

NOTE

If an interrupt request occurs during the execution of an SMK instruction disabling that interrupt, the interrupt may or may not be accepted (depending on the exact timing of the interrupt signal with respect to the execution of the SMK instruction); therefore, the interrupt mask register should be changed only when interrupt is inhibited.

Table 4-8.
Priority Interrupt Mask Assignments

A-Register Bit No.	SMK '0120	SMK '0220	SMK '0320	Interrupt Line Number
1	1	17	33	
2	2	18	34	
3	3	19	35	
4	4	20	36	
5	5	21	37	
6	6	22	38	
7	7	23	39	
8	8	24	40	
9	9	25	41	
10	10	26	42	
11	11	27	43	
12	12	28	44	
13	13	29	45	
14	14	30	46	
15	15	31	47	
16	16	32	48	

MEMORY INCREMENT OPTION, MODEL 416-26

Groups of four Priority Interrupt Lines may be optionally changed to Memory Increment Break lines. Any number of Priority Interrupt Groups may be so modified; however, the modified groups must be consecutive starting with the first group of four lines.

The function performed by a Memory Increment Break is:

$$[\text{Ded. Location}] + 1 \rightarrow [\text{Ded. Location}]$$

There is no overflow indication and no interrupt generated on overflow. Execution of the break requires three cycles.

NOTE

Memory increment requests are not subject to control by the INH or ENB instructions; however, mask register bits are associated with memory increment lines for individual line control as described under Priority Interrupt Control, above.

MEMORY ACCESS PRIORITY STRUCTURE

An access-to-memory priority structure is built into the basic computer control logic; the priority assignments are given in Table 4-9. The functions listed in Table 4-9, with the exception of the CPU, may be categorized in either of the following groups.

1. Program Interrupt - those functions which interrupt the normal sequence of instructions being performed by altering the P counter.
2. Computer Break - those functions which interject a function without altering the P counter.

Table 4-9.
DDP-416 Computer Access-to-Memory Priority Structure

Relative Priority Level	Option/Function
1	Direct Memory Access Break (DMA) 416-21
2	Direct Multiplex Control Break (DMC) 416-20
3	Power Failure Interrupt (PFI), Standard
4	Real Time Clock Break 416-12
5	Memory Lockout Violation Interrupt 416-08
6	Standard Interrupt, Standard
7	Memory Increment Break 416-26
8	Priority Interrupt 416-25
9	Central Processing Unit (CPU)

Program Interrupts

The program interrupt group includes (1) SI - Standard Interrupt, (2) PFI - Power Failure Interrupt, (3) PI - Priority Interrupt, and (4) ML - Memory Lockout Violation Interrupt. Interrupts of this type can occur only when the CPU has completed an instruction; the interrupt is accomplished by forcing the CPU to generate an indirect JST instruction to a dedicated location.

NOTE

A standard interrupt or a priority interrupt may only occur when the CPU is in the "permit interrupt" status; Memory Lockout Violation Interrupt and Power Failure Interrupt may occur regardless of the CPU "permit interrupt" status.

Computer Breaks

The computer break group includes (1) RTC - Real Time Clock Break, (2) MI - Memory Increment Break, (3) DMC - Direct Multiplex Control Break, and (4) DMA - Direct Memory Access Break. Real Time Clock, Memory Increment and DMC breaks can occur only when the CPU has completed an instruction. DMA breaks, however, may occur without waiting for the end of an instruction. All breaks may occur independent of the "permit interrupt" status.

SECTION V PERIPHERAL DEVICES

ASR-33/35 TELETYPE UNITS, MODEL 416-53/55

The ASR-33/35 Teletype unit is available as a basic I/O device with the DDP-416. The ASR-33/35 is a versatile device providing a capability to read paper tape at 10 characters/second, and punch paper tape at the same rate. The ASR-33/35 may also print out data from the computer at 10 characters/second and transfer data to the computer from the keyboard. In the local mode, the unit may be used for off-line paper tape preparation, reproduction and listing.

Keyboard and Carriage Features

The ASR-33/35 keyboard is similar to that of a standard typewriter. The keyboard includes four rows of keys and generates an eight-level code. Letters and numerals are transmitted without a shift, similar to lower-case transmission on a typewriter. Printing characters (?, =, *, etc.) are typed by using the shift key, similar to upper-case positions on certain typewriter keys. Control functions, generated using the control (CTRL) key, are X-OFF (S-key), X-ON (Q-key), EOM (C-key) and BELL (G-key). The LINE FEED and RETURN codes are transmitted without the CTRL key being depressed.

The ASR-33/35 is capable of printing a 72/75-character line. If a programmer wishes to print 71/74 or fewer characters, he must perform a carriage return and line feed (in that order) after the last character desired in each line.

Keyboard Interlock. -- The ASR-33/35 keyboard is interlocked for all keys except the SHIFT, CTRL and REPT keys, preventing more than one key from being depressed at a time. The keyboard does not lock in the upper-case position. Therefore, the operator must hold the SHIFT key depressed to produce upper-case characters.

Operating Modes, ASR-35 Unit

The ASR-35 may be operated in either an on-line or an off-line mode. The modes are described in the following paragraphs.

Off-Line. -- The ASR-35 off-line modes are as follows.

- 1) K Mode - keyboard to printer (ASCII)
- 2) KT Mode - (a) Keyboard to printer and punch (ASCII)
(b) Reader to printer and punch (ASCII or binary)
- 3) T Mode - (a) Keyboard to punch (ASCII)
(b) Reader to printer (ASCII or binary)

On-Line. -- The ASR-35 on-line modes are as follows.

- 1) K Mode - (a) Input transfer from the keyboard monitored by the printer (ASCII)
(b) Output transfer to the printer (ASCII)
- 2) KT Mode - (a) Input transfer from the keyboard monitored by the printer and the punch* if enabled (ASCII)
(b) Input transfer from the reader † monitored by the printer and the punch* if enabled (ASCII or binary)
(c) Output transfer to the printer and punch* if enabled (ASCII or binary)
- 3) T Mode - (a) Input transfer from the reader † monitored by the printer (ASCII or binary)
(b) Output transfer to the printer (ASCII)
(c) Simultaneous off-line operation of keyboard to punch (ASCII)
- 4) TTS Mode - (a) Input transfer from the reader (any 8 level code). Manual control of reader only. Automatic start and stop code inoperative
(b) Simultaneous off-line operation of keyboard to punch (ASCII)
- 5) TTR Mode - Output transfer to the punch (any 8 level code)

† See Reader Control

* See Punch Control

Reader Control (Effective in KT and T Modes Only)

Starting. -- To start the reader under program control, the program must output an X-ON character; $(021)_8$ or $(221)_8$. After waiting until the ASR is not busy, OCP the ASR in input mode and proceed with input transfer instructions. The reader may also be started by depressing the manual reader switch to the start position. After the reader is started, the first character to be read is the one initially positioned over the read pins. The reader will stop by detecting an X-OFF character if it has been started manually.

Stopping. -- The reader will stop, when operating under program control, by recognition of an X-OFF character; $(023)_8$ or $(223)_8$. The X-OFF character will be read into the character buffer and the two following characters on tape will be read into the buffer, unless the character following X-OFF is RUB-OUT, in which case only RUB-OUT is read into the buffer, before the tape stops. The reader may not be stopped manually by depressing the reader switch to the stop position if it was started by transmission of X-ON. A stop code will stop the reader off-line.

On-Line Punch Control

This is effective in the KT mode only.

Punch On. -- The computer must output a TAPE character $(022)_8$ or $(222)_8$ which will enable the punch. A BUFFER RUB-OUT character or a time delay equal to one character time must follow the TAPE character. Neither the tape nor the RUB-OUT character will be punched on the tape. The following output transfers will then be punched on the tape.

Punch Off. -- The computer must output an X-OFF character $(023)_8$ or $(223)_8$ to disable the punch. The X-OFF character must be followed by a RUB-OUT character $(377)_8$ or an equivalent time delay. Both characters will be punched on the tape.

Off-Line (Local) Punch Control

The punch will be enabled in the KT and T modes and disabled in all other modes. Punch control is not effective off-line.

Tape Leader. -- Tape Leader may be generated off-line (local) by depressing the BREAK button until the required amount of leader is punched. The operator should then depress local backspace and then depress rub-out. This will present an all ZEROS leader with the last character being a rub-out; or all ONES delete character.

Operating Modes - ASR-33 Unit

Tape Reader

Starting. -- The reader is started under program control as follows.

- a. Enable the ASR-33 in the output mode using OCP 104.
- b. Output an X-ON character $(221)_8$ using OTA 004.
- c. Delay while the ASR-33 is busy (test with SKS 104)
- d. Enable the ASR-33 in the input mode using OCP 004.

To use this method, the ASR-33 must first be set up in the output mode by an OCP. After the X-ON character to the ASR-33 buffer is outputted by an OTA, the SKS not busy test must follow. When the not busy indication is obtained, the ASR-33 must then be OCP'd in Input Mode; whereupon INAs can then be executed. Manual starting is controlled by the START/STOP switch. After the reader is started, the first character to be read is the one initially positioned over the read pins.

Stopping. -- The reader stops automatically only when an "X-OFF" code $(223)_8$ or $(023)_8$, is read from paper tape. The X-OFF character will be transmitted into the device's buffer and the character following will be transmitted into the buffer before the reader stops. Manual stops are controlled by the START/STOP switch. (The reader also stops automatically when it runs out of paper tape.)

Overriding Stop Code. -- A stop code will stop the reader while tape is being duplicated off line. To continue duplicating, manually restart the reader with the START switch.

Tape Punch

The punch is controlled by manual operation of the punch ON-OFF switch located on the ASR-33. When the punch is on, any input from or output to the ASR-33 will cause tape to be punched. Tape leader may be generated in bursts of 20 sprockets with each depression of the HERE-IS key.

Off-Line Operation

Off-line operation of the ASR-33 includes the following data transmission.

- a. Keyboard to printer
- b. Keyboard to printer and punch
- c. Reader to printer
- d. Reader to printer and punch

ASR-33/35 On-Line Operating Modes

There are two basic modes of operation for the ASR-33/35 when on line: input mode and output mode. These are set up by the appropriate OCP instruction. Once set up, the ASR-33/35 remains in a given mode until it is changed by another OCP.

Input Mode. -- The input mode is used to transmit information from the ASR-33/35 keyboard to the computer or from the reader to the computer. In either case, printed copy is produced if the 8-bit character is printable, and a control function is performed if the 8-bit character is a control character (see Appendix C). If characters are being read from the reader, any of the 256 possible 8-bit characters appearing on the tape will be transmitted to the computer. When an X-OFF, $(223)_8$, or $(023)_8$ is read, the reader will stop after reading the character following the X-OFF, unless that following character is an X-ON $(221)_8$ or $(021)_8$.

Output Mode. -- The output mode is used to transmit information from the computer to the ASR-33/35 printer or the printer and the punch. In either case, printed copy is produced if the 8-bit character is printable, and a control function is performed if the 8-bit character is a control character. When punching, any 8-bit code (of the possible 256) transmitted from the computer will be punched whether it is printable or not. Certain 8-bit codes -- $(221)_8$, $(021)_8$, $(005)_8$ and $(205)_8$ -- when transmitted from the computer will also cause a control action by the ASR-33/35 and prevent proper transmission of further characters. X-ON, $(221)_8$ or $(021)_8$ will start the paper tape reader, and on the ASR-33 WRU $(205)_8$ or $(005)_8$ will trigger the answer-back drum.

Character Modes

Within either the input or output modes, either of two character modes, ASCII or binary, may be used. Code type is selected by individual INA or OTA instructions and may be intermixed in any manner (though this is not normally done).

ASCII Mode. -- In the ASCII mode a full 8-bit character is transmitted to or from the least significant 8 bits of the A-register and the ASR-33/35. This permits transmission of any standard character or control character from the reader or keyboard of the ASR-33/35 to the computer or from the computer to the printer or punch of the ASR-33/35.

Binary Mode. -- In the binary mode a 6-bit character is transmitted to or from the least significant 6 bits of the A-register and the ASR-33/35. In the case of output in the binary mode, an additional 2 bits are automatically added in the high-order position to the 6-bit character to form an 8-bit character acceptable to the ASR-33/35. The 2 bits added are chosen so that the resulting 8-bit character is an alphanumeric character, not a control character. On input, the two high-order bits of the 8-bit character transmitted by the ASR-33/35 are stripped and ignored.

Instructions

The following instructions are used to control the ASR-33/35 and to transfer data to and from it.

OCP '0004 Enable ASR-33/35 in Input Mode

This instruction sets up the device interface to accept characters from the ASR-33/35. It should be given any time it is desired to switch the ASR-33/35 from the output mode to the input mode. This instruction must not be given while the ASR-33/35 is busy. Thus, an SKS "not busy" test should precede the instruction.

OCP '0104 Enable ASR-33/35 in Output Mode

This instruction sets up the device interface to transmit characters to the ASR-33/35. The instruction must be given any time it is desired to switch from the input to the output mode. The instruction must not be given while the ASR-33/35 is busy. Thus, an SKS "not busy" test should precede the instruction.

SKS '0404 Skip if ASR-33/35 Is Not Interrupting

This instruction tests whether the ASR-33/35 has caused an interrupt on the standard interrupt line.

SKS '0004 or '0204 Skip If ASR-33/35 Is Ready

This instruction tests whether the ASR-33/35 device interface is ready to accept another character from the computer or to present another character to the computer.

SKS '0104 Skip If ASR-33/35 Is Not Busy

The ASR-33/35 busy signal is defined as follows.

a. In the output mode the ASR-33/35 is busy from the time a character is transmitted from the computer to the ASR-33/35 device interface until it has been serially shifted out to the ASR-33/35. This time is approximately 105 ms.

b. In the input mode the ASR-33/35 is busy from the time the ASR-33/35 starts to serially transfer a character to the device interface until the transfer is complete and the ASR-33/35 ready condition is present. This time is approximately 100 ms.

SKS '0504 Skip If Stop Code Was Not Read on ASR-33/35

This instruction tests whether a stop code (223)₈ or (023)₈ has been read on the ASR-33/35. The stop code indication can be tested as soon as the stop code has been read from the ASR-33/35 into the device buffer and is ready for input to the computer. When a

stop code is read by an ASR-33/35, the stop code and ONE/TWO following characters will be transferred to the device buffer before the reader stops. The stop code indication will remain present until the character following the stop code is ready for input to the computer (approximately 100 ms).

INA '0004 Input in ASCII Mode If Ready*

This instruction transmits the full 8-bit character from the ASR-33/35 to the 8 least significant bits of the A-register. The A-register is not cleared.

INA '0204 Input in Binary Mode If Ready*

This instruction transmits the 6 least significant bits of the 8-bit ASR-33/35 character to the 6 least significant bits of the A-register. The A-register is not cleared.

INA '1004 Clear A and Input in ASCII Mode If Ready*

INA '1204 Clear A and Input in Binary Mode If Ready*

OTA '0004 Output in ASCII Mode If Ready

This instruction transmits the 8 least significant bits of the A-register to the ASR-33/35. If the ASR-33/35 is punching, it will punch all 8 bits of the code that is transmitted. However, in printing, it will determine the character to be printed or the control function to be performed from the 7 least significant bits.

OTA '0204 Output in Binary Mode If Ready

This instruction transmits the 8 least significant bits of the A-register to the ASR-33/35 and then modifies channel 7 (normally A10) to form a valid ASCII alphanumeric character. To do this, bit 7 is made the inverse of A11. Thus, if the 8 least significant bits in the A-register were $(XX1XXXXX)_2$, they would be transmitted to the ASR-33/35 as $(X01XXXXX)_2$. If they were $(XX0XXXXX)_2$, they would be transmitted as $(X10XXXXX)_2$.

Standard Interrupt

The OTB mask bit assignment for standard interrupt is OTB 11.

Input Mode. -- An interrupt request will occur when data is in the buffer and Ready is set. When the interrupt is honored by executing an INA and data is transferred, the controller will not be busy and the interrupt request will be reset.

*READY must be honored within one millisecond to ensure taking the character.

Output Mode. -- An interrupt request will occur whenever the Ready Flip-Flop is set (controller ready to accept data from the CPU). The request can be reset by executing an OTA or OCP input command.

Paper Tape Format And ASR Codes

The format of the ASR-33/35 paper tape is shown in Figure 5-1. The codes for the ASR-33/35 characters and symbols are shown in Table 5-1.

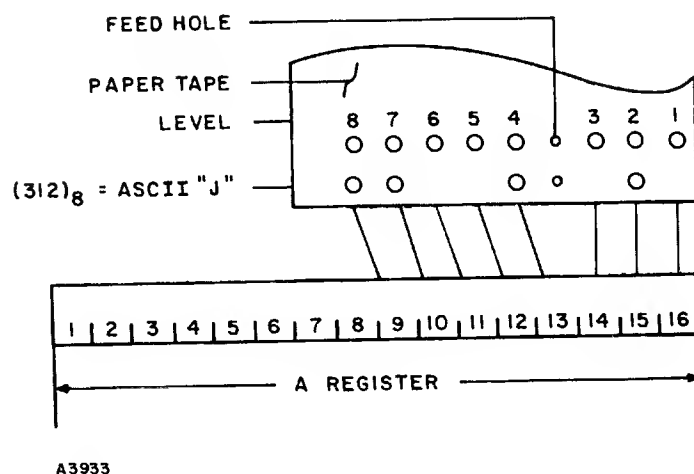


Figure 5-1. ASR-33/35 Paper Tape Format

Table 5-1
ASR-33/35 Characters and Symbol Codes

KEY	Lower Case Code	S Y M	Shift Code	M	Control Code	S Y M
0	260	0	LO		260	0
1-!	261	1	241	!	261	1
2-"	262	2	242	"	262	2
3-#	263	3	243	#	263	3
4-\$	264	4	244	\$	264	4
5-%	265	5	245	%	265	5
6-&	266	6	246	&	266	6
7-'	267	7	247	'	267	7
8-(270	8	250	(270	8
9-)	271	9	251)	271	9
A	301	A	LO		201	
B	302	B	LO		202	
C	303	C	LO		203	
D-EOT	304	D	LO		204	EOT
E-WRU	305	E	LO		205	WRU
F-RU	306	F	LO		206	RU
G-BELL	307	G	LO		207	BELL
H	310	H	LO		210	
I-TAB	311	I	LO		211	TAB
J	312	J	LO		212	LF
K-VT	313	K	333	[213	
L-FORM	314	L	334	\	214	
M	315	M	335]	215	CR
N-↑	316	N	336	↑	216	
O-←	317	O	337	←	217	
P-@	320	P	300	@	220	
Q-XON	321	Q	LO		221	XON
R-TAPE	322	R	LO		222	TAPE
S-XOFF	323	S	LO		223	XOFF
T-TAPE	324	T	LO		224	TAPE
U	325	U	LO		225	
V	326	V	LO		226	
W	327	W	LO		227	
X	330	X	LO		230	
Y	331	Y	LO		231	
Z	332	Z	LO		232	
:-*	272	:	252	*	272	:
-(minus)-=	255	-	275	=	255	-
* HERE IS	000		020		000	
ALT MODE	375		LO		275	=
LF	212		LO		212	
CR	215		LO		215	
; ~ +	273	;	253	+	273	;
RUB OUT	377		LO		277	?
BREAK	NOTE					
, - <	254	,	274	<	254	,
. - >	256	.	276	>	256	.
/ - ?	257	/	277	?	257	/
SPACE	240		LO		240	

* Applicable for ASR-33 only.

Note: While BREAK is depressed, a 000 code will be generated.
When BREAK is released, an indeterminate character will be produced.

LO: Locked Out

HIGH-SPEED PAPER-TAPE READER, MODEL 416-50

A unidirectional perforated-tape reader (Figure 5-2) is available as an option. The unit reads eight data channels plus a sprocket hole channel, at the rate of 30 in./sec (10 characters per inch, or 300 characters per second).

The unit uses standard paper or mylar tapes (black paper recommended) 0.004 to 0.005 in. thick. The tape can be loaded without removing power by rotating a front-mounted READY-LOAD switch clockwise to the LOAD position. After the tape has been loaded, the READY-LOAD switch must be rotated counterclockwise to the READY position.

Reader Modes

The high-speed paper-tape reader operates in the continuous mode at a rate of 300 characters per second. Reader start is initiated by an OCP command, after which the reader will continuously transfer characters to its buffer until the complete tape has been read or an OCP stop-reader command is executed.

Codes

Eight-level code will be read into the device buffer. The reader is passive and will transfer all eight-level bit configurations into the computer (including tape leader). The reader hardware will not interpret any code as a stop or delete code.

Specifications

The paper-tape reader has the following characteristics:

- a. Reads at speed up to 300 cps
- b. Start time 5 ms
- c. Stop on next character at 300 cps
- d. Reads paper or mylar tape 1 in. wide and 0.004 to 0.005 in. thick with transmissivity of up to 40%.
- e. Density of 10 characters per inch.

Commands

The following instructions are used for communication between the reader and the computer:

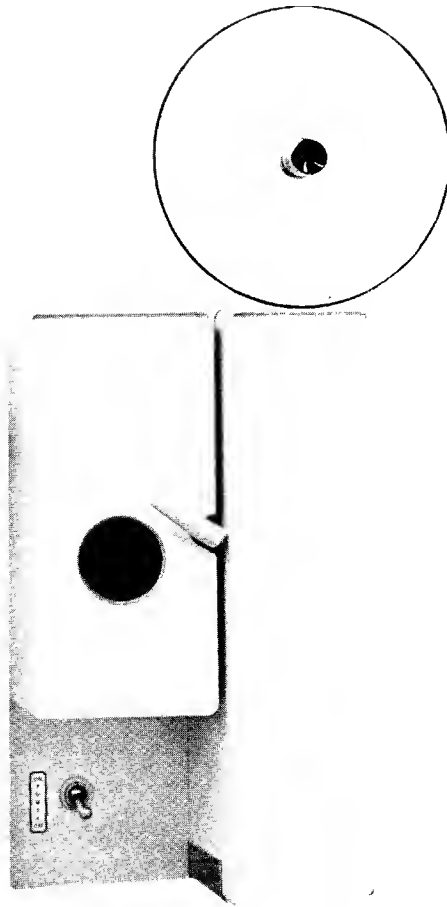


Figure 5-2.
High-Speed Paper-Tape Reader, Model 416-50

OCP '0001 Start Reader

This instruction starts tape motion. The first character to pass the read station is transferred to the interface buffer for transmission to the computer.

OCP '0101 Stop Reader

This instruction stops tape motion. In order to avoid losing a character after a restart, this OCP must be executed within 1.0 ms of a character-ready signal.

SKS '0401 Skip If Tape Reader Not Interrupting

The tape reader is interrupting when a character is ready and the interrupt mask flip-flop is set.

SKS '0001 Skip If Tape Reader Ready

The tape reader is ready when a character is available in the interface buffer.

INA '0001 Input From Paper Tape Reader If Ready

Execution of this instruction causes a character to be ORed into the eight least significant bits of the A-register and causes the next program instruction to be skipped.

INA '1001 Clear A-Register and Input From Paper Tape Reader If Ready

This instruction is identical to INA '0001, but the A-register is cleared before the character is transferred in.

SMK '0020 Set Interrupt Masks

This instruction sets the mask bit if the corresponding bit of the A-register is a ONE and resets the mask bit if the corresponding bit is a ZERO. The instruction is unconditional and never skips. A-register bit 09 controls the paper-tape reader interrupt mask flip-flop.

HIGH-SPEED PAPER-TAPE PUNCH, MODEL 416-52

The paper tape punch (Figure 5-3) is capable of punching 1-inch, 8-level tape at a rate of up to 110 characters per second. Power to the punch is normally controlled by the computer program. After a power-on command is initiated, the interface prevents data transfers for a 5-second delay period to allow the punch mechanism to come up to full operating speed. Punch power may also be controlled by a manual switch on the paper tape equipment cabinet. This switch facilitates maintenance and replacement of the tape supply. However, it should not be used for normal on-line operation.

The punch is a synchronous unit; pulses generated by a magnetic pickup coil synchronize the interface control circuits.

Eight levels of data are transferred to the punch from the computer output bus, bits 9 through 16. Tape tracks 1 through 8 represent output bus bits 16 through 9, respectively.

Specifications

The Model 416-52 Paper-Tape Punch has the following characteristics:

- a. Punches oiled paper, foil or mylar tape.
- b. Paper supply of 1000 feet.
- c. Tape width of 1.000 in.
- d. Density of 10 characters per in.

Loading Procedure

The tape should be threaded off the bottom of the roll through the wire and roller guides to the tape guide and punch block. The tape lid is then lifted and tape led between the lid and the feed wheel and out under the tape cutter. Punch power is then turned on and the feedout lever (located at the top center of the punch cover) is depressed while pulling the tape to the left. When the tape begins to feed, the feed lever may be released.

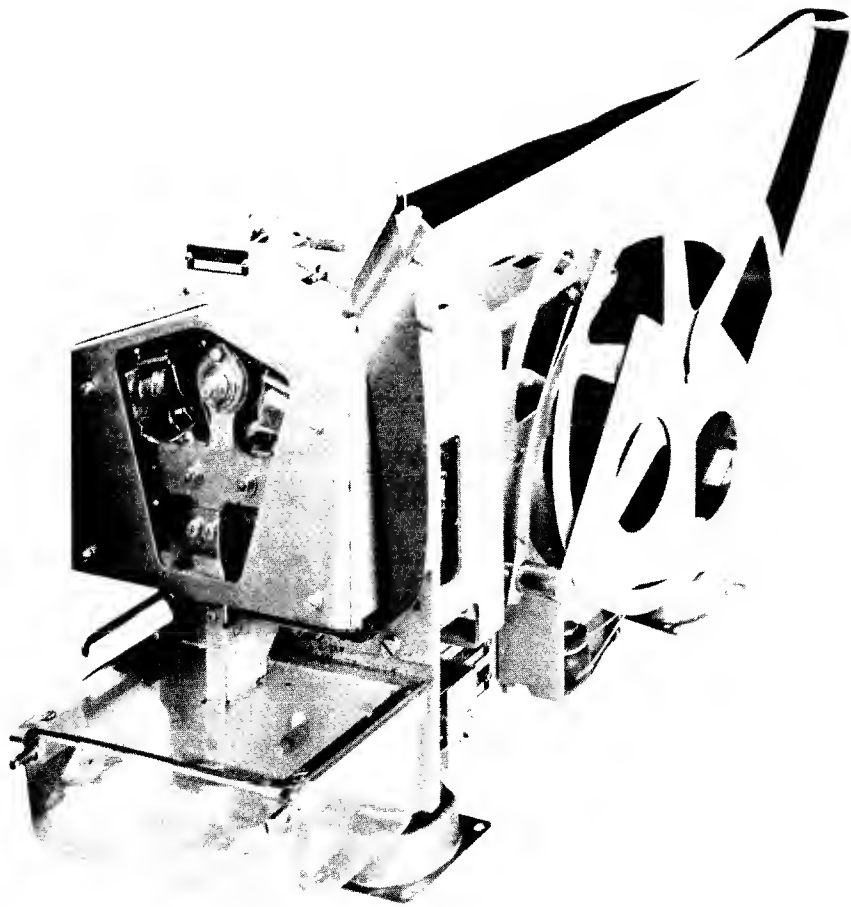


Figure 5-3.
High-Speed Paper-Tape Punch, Model 416-52

Commands

The following instructions are used for communication between the punch and the computer.

OCP '0002 Enable Paper Tape Punch

This instruction turns on power to the punch. There is a 5-second delay before the punch is ready to accept a character.

OCP '0102 Turn Punch Power Off

This instruction removes punch power. Before executing this instruction, however, the punch must be sensed for a ready condition to avoid turning the punch off while a character is being punched.

SKS '0002 Skip If Paper Tape Punch Is Ready

The punch is ready when its buffer is empty and ready to accept new data from the computer.

SKS '0402 Skip If Paper Tape Punch Is Not Interrupting

The tape punch is interrupting when its interrupt mask flip-flop is set and its buffer is ready to receive a character.

SKS '0102 Skip If Punch Power Is On

This instruction must precede an OCP '0002. If the latter is executed when power is already enabled, the loss of a character might occur.

OTA '0002 Output to Tape Punch If Ready

This instruction performs the output transfer; if ready, the 8 least significant bits of the A-register are transferred to the punch interface, the next instruction is skipped, and a punch cycle is started. (During a punch cycle, the interface responds to this instruction or the SKS '0002 as not ready.)

SMK '0020 Set Interrupt Masks

This instruction sets a mask bit if the corresponding bit of the A-register is a ONE and resets the mask bit if the corresponding bit is a ZERO. The instruction is unconditional and never skips. The paper tape punch interrupt mask flip-flop is controlled by A-register bit 10.

SECTION VI SAMPLE PROGRAMS

Key-In Loader for ASR-33/ ASR-35

```

*   TO LOAD PAL-MODE PROGRAMS, THE FOLLOWING PROCEDURE
*   MUST BE FOLLOWED.
*   1. IF THE KEY-IN LOADER IS NOT PRESENT IN LOCATIONS
*      1-17 OCTAL, MANUALLY KEY IN THE FOLLOWING:
*
*           ASR          DIGITRONICS
*   1 STA  *57          010057    010057
*   2 OCP  *0001/4      030004    030001
*   3 INA  *1001/4      131004    131001
*   4 JMP  *-1          002003    002003
*   5 SNZ          101040    101040
*   6 JMP  *-3          002003    002003
*   7 STA  0            010000    010000
*   10 INA  *1001/4      131004    131001
*   11 JMP  *-1          002010    002010
*   12 LGL  8            041470    041470
*   13 INA  *0001/4      130004    130001
*   14 JMP  *-1          002013    002013
*   15 STA*  0            110000    110000
*   16 IRS  0            024000    024000
*   17 SZE          100040    100040
*
*   2. MASTER CLEAR
*   3. SET P REGISTER TO 1
*   4. MOUNT PAL-MODE TAPE IN INPUT DEVICE AND PRESS START

```

Fixed Point, Double Precision Add Subroutine

```

*           THE MOST SIGNIFICANT HALF OF THE DOUBLE PRECISION
*           SUM WILL BE IN THE SIMULATED A REGISTER. THE
*           LEAST SIGNIFICANT HALF WILL BE IN THE SIMULATED
*           B REGISTER.
*           THE REAL A REGISTER WILL CONTAIN THE MOST
*           SIGNIFICANT HALF OF THE DOUBLE PRECISION SUM
*           UNLESS IT IS EQUAL TO ZERO IN WHICH CASE
*           THE REAL A REGISTER WILL CONTAIN THE
*           LEAST SIGNIFICANT HALF.

DADD DAC    **-*          DOUBLE ADD SUBROUTINE
LDA* DADD    FORM THE
JST ARGX     ARGUMENT ADDRESS
LDA* ADDR    FETCH THE ARGUMENT WITH SIGN
ERA AREG     FORM THE HALF ADD OF SIGNS
STA SIGN     SAVE FOR OVERFLOW CONTROL
ERA AREG     RECOVER THE ARGUMENT
STA ARG1     SAVE FOR OVERFLOW CONTROL
ADD AREG     ADD THE HIGH PARTS
STA AREG     AND STORE THE RESULT
IRS ADDR     STEP TO LOW ORDER BITS
LDA* ADDR    FETCH THE LOW BITS
ADD BREG     ADD THE LOW ORDER BITS
SPL          TEST FOR A CARRY
IRS AREG     PROPAGATE THE CARRY
NOP          TO HIGH ORDER WORD
ANA MAG      ERASE ANY CARRY
STA BREG     SAVE THE LOW ORDER SUM
LDA SIGN     TEST FOR
SPL          POSSIBLE OVERFLOW
JMP ADD1     SIGNS DIFFER-NO OA POSSIBLE
LDA ARG1     SIGNS ALIKE - TEST FOR
ERA AREG     A SIGN CHANGE
SPL          X
JST DOAS     A SIGN CHANGE - OVERFLOW
ADD1 JST ENDX SETUP THE EXIT CONDITIONS
IRS DADD     STEP TO THE RETURN
JMP* DADD    AND EXIT

```

Fixed Point, Double Precision Subtract Subroutine

* THE TWO WORD DOUBLE PRECISION MINUEND (XH,XL)
 * IS FOUND IN THE SIMULATED A REGISTER (AREG)
 * AND B REGISTER (BREG). AREG CONTAINS THE
 * MOST SIGNIFICANT HALF OF THE MINUEND,
 * BREG CONTAINS THE LEAST SIGNIFICANT HALF
 * OF THE MINUEND.
 * THE ADDRESS OF THE MOST SIGNIFICANT HALF
 * OF THE SUBTRAHEND (YH) IS FOUND IN THE
 * NEXT SEQUENTIAL LOCATION AFTER THE CALL.
 * THE LEAST SIGNIFICANT HALF OF THE SUBTRAHEND
 * (YL) IS FOUND IN LOCATION YH+1
 * THE DOUBLE PRECISION DIFFERENCE WILL BE FOUND
 * IN THE SIMULATED A AND B REGISTERS. THE MOST
 * SIGNIFICANT HALF WILL BE IN AREG, THE LEAST
 * SIGNIFICANT HALF WILL BE IN BREG.

DSUB	DAC	*--*	DOUBLE SUBTRACT SUBROUTINE
	LDA*	DSUB	FORM THE DIRECT
	JST	ARGX	ARGUMENT ADDRESS IN ADDR
	LDA	AREG	SAVE THE HIGH ORDER A-REGISTER
	STA	ARG1	BITS FOR OVERFLOW DETECTION
	ERA*	ADDR	FORM THE SIGN CONTROL WORD
	STA	SIGN	AND SAVE IT
	ERA	AREG	RECOVER THE SUBTRAHEND
	SUB	AREG	FORM S-A=-(A-S)
	STA	AREG	AND SAVE IT
	IRS	ADDR	STEP TO THE LOW ORDER WORD
	LDA	BREG	FORM THE LOW PART OF A-S
	SUB*	ADDR	X
	SPL		TEST FOR A BORROW
	IRS	AREG	A BORROW - FORM (S-A+1)
	NOP		X
	ANA	MAG	FORM THE 15 LOW BITS ONLY
	STA	BREG	SAVE THE LOW ORDER RESULT
	CRA		FORM
	SUB	AREG	(A-S) OR (A-S-1) IF A BORROW
	STA	AREG	AND SAVE AS HIGH ORDER RESULT
	ERA	ARG1	TEST FOR A SIGN CHANGE
	SMI		X
	JMP	SUB1	NO SIGN CHANGE - NO OVERFLOW
	ERA	SIGN	IS SIGN CHANGE OVERFLOW
	SMI		TEST FOR DIFFERENT SIGNS
	JST	DOAS	SIGNS DIFFERED - OVERFLOW
SUB1	JST	ENDX	SETUP EXIT CONDITIONS
	IRS	DSUB	STEP TO THE RETURN
	JMP*	DSUB	AND EXIT

Fixed Point, Double Precision Multiply Subroutine

```

*          THE DOUBLE PRECISION MULTIPLICAND (XH,XL) IS
*          FOUND IN THE SIMULATED A REGISTER (AREG)
*          AND B REGISTER (BREG). AREG CONTAINS THE
*          MOST SIGNIFICANT HALF OF THE MULTIPLICAND,
*          BREG CONTAINS THE LEAST SIGNIFICANT HALF.
*          THE ADDRESS OF THE MOST SIGNIFICANT HALF
*          OF THE MULTIPLIER (YH) IS FOUND IN THE NEXT
*          SEQUENTIAL LOCATION AFTER THE CALL. THE
*          LEAST SIGNIFICANT HALF (YL) IS FOUND IN
*          LOCATION YH+1
*
*          THE MOST SIGNIFICANT HALF OF THE EXTENDED
*          PRECISION PRODUCT WILL BE IN THE SIMULATED
*          A REGISTER. THE LEAST SIGNIFICANT HALF WILL
*          BE IN THE SIMULATED B REGISTER.
DMPY DAC    **--      DOUBLE MULTIPLY SUBROUTINE
      LDA    AREG      SAVE THE HIGH ORDER
      STA    SIGN      A-REGISTER FOR SIGN CONTROL
      SPL                    TEST FOR POSITIVE A-REGISTER
      JST    DTCA      NEGATIVE-TWOS COMPLEMENT
      LDA    AREG      SAVE THE DOUBLE LENGTH
      STA    ARG3      A-REGISTER AS THE
      LDA    BREG      MULTIPLICAND
      STA    ARG4      X
      LDA*   DMPY      MOVE THE ARGUMENT ADDRESS
      STA    **2      TO THE CALLING SEQUENCE
      JST    DLDA      LOAD THE MULTIPLIER INTO
      DAC    **--      THE DOUBLE ACCUMULATOR
      ERA    SIGN      FORM THE SIGN OF THE PRODUCT
      STA    SIGN      AND SAVE IT
      LDA    AREG      FORM THE
      SPL                    MAGNITUDE OF
      JST    DTCA      THE DOUBLE ACCUMULATOR
      CPA                    CLEAR THE
      STA    ARG1      PRODUCT REGISTER
      STA    ARG2      X
      LDA    M15      SET THE COUNT FOR
      STA    KNTR      15 BITS
      LDA    BREG      USE THE LOW ORDER BITS
      JST    XMPY      FOR THE INITIAL MULTIPLIER
      LDA    M16      SET THE COUNT FOR
      STA    KNTR      16 BITS
      LDA    AREG      USE THE HIGH ORDER BITS
      JST    XMPY      AS THE FINAL MULTIPLIER
      LDA    ARG2      MOVE THE
      STA    BREG      PRODUCT TO THE
      LDA    ARG1      DOUBLE LENGTH
      STA    AREG      A-REGISTER
      LDA    SIGN      TEST FOR A
      SPL                    NEGATIVE RESULT
      JST    DTCA      TWOS COMPLEMENT THE PRODUCT
      LDA    SIGN      CHECK FOR
      ERA    AREG      (-1)*(-1)=1
      SPL                    X
      JST    NOAS      THIS IS IT, OVERFLOW
      JST    ENDX      SETUP THE END CONDITIONS
      IRS    DMPY      STEP TO THE RETURN
      JMP*   DMPY      AND EXIT

```

*

Fixed Point, Double Precision Multiply Subroutine (Cont)

XMPY	DAC	***	N BY 30 BIT MULTIPLY ROUTINE
	STA	MPYR	SAVE THE MULTIPLIER
SHFT	LDA	ARG1	RIGHT SHIFT
	ARR	1	THE PARTIAL
	STA	TEMP	PRODUCT ONE
	ANA	MAG	BIT POSITION
	STA	ARG1	X
	ERA	TEMP	X
	ERA	ARG2	X
	LGR	1	X
	STA	ARG2	X
	LDA	MPYR	ROTATE THE
	ARR	1	MULTIPLIER
	STA	MPYR	ONE POSITION
	SMI		TEST THE OLD LOW BIT
	JMP	MPY1	A ZERO- NO ADDITION
	LDA	ARG2	ADD THE MULTIPLICAND
	ADD	ARG4	TO THE PARTIAL PRODUCT
	SPL		X
	IRS	ARG1	X
	ANA	MAG	X
	STA	ARG2	X
	LDA	ARG1	X
	ADD	ARG3	X
	STA	ARG1	X
MPY1	IRS	KNTR	COUNT THE MULTIPLIER BITS
	JMP	SHFT	STILL MORE TO GO
	JMP*	XMPY	ALL DONE - EXIT

Fixed Point, Double Precision Divide Subroutine

```

*           THE DOUBLE PRECISION DIVIDEND (XH,XL) IS FOUND
*           IN THE SIMULATED A REGISTER (AREG) AND B
*           REGISTER (BREG). AREG CONTAINS THE MOST
*           SIGNIFICANT HALF OF THE DIVIDEND, BREG CONTAINS
*           THE LEAST SIGNIFICANT HALF.
*           THE ADDRESS OF THE MOST SIGNIFICANT HALF
*           OF THE DIVISOR (YH) IS FOUND IN THE NEXT
*           SEQUENTIAL LOCATION AFTER THE CALL. THE
*           LEAST SIGNIFICANT HALF (YL) IS FOUND IN
*           LOCATION YH+1.
*           THE MOST SIGNIFICANT HALF OF THE QUOTIENT
*           WILL BE IN THE SIMULATED A REGISTER. THE
*           LEAST SIGNIFICANT HALF WILL BE IN THE
*           SIMULATED R REGISTER.
DDIV DAC  **-*          DOUBLE DIVIDE SUBROUTINE
      LDA  AREG          SAVE THE INITIAL A-REGISTER
      STA  SIGN          SIGN FOR SIGN CONTROL
      SPL                      IF NEGATIVE - FORM THE
      JST  DTCA          MAGNITUDE OF THE DIVIDEND
      SPL                      TEST FOR -1 AS DIVIDEND
      JMP  DCK          -1/X IS DIVIDE CHECK
      LDA  AREG          MOVE THE DIVIDEND
      STA  ARG1          TO WORKING STORAGE
      LDA  BREG          X
      STA  ARG2          X
      LDA* DDIV          MOVE THE ARGUMENT ADDRESS
      STA  **+2          TO THE CALLING SEQUENCE
      JST  DLDA          LOAD THE DIVISOR INTO
      DAC  **-*          THE DOUBLE LENGTH ACCUMULATOR
      ERA  SIGN          FORM THE QUOTIENT SIGN
      STA  TEMP          AND SAVE IT
      LDA  AREG          TEST FOR
      SPL                      NEGATIVE DIVISOR
      JST  DTCA          FORM MAGNITUDE OF DIVISOR
      SPL                      TEST FOR DIVISION BY -1
      JMP  DIVY          X/-1 IS -X
      LDA  ARG1          TEST FOR
      SUB  AREG          DIVIDE CHECK
      SPL                      X
      JMP  DIV1          OK - DIVISOR IS GREATER
      SZE                      TEST FOR DIVIDEND GREATER
      JMP  DCK1          DIVIDEND GREATER - DIVIDE CHECK
      LDA  ARG2          HIGH PARTS EQUAL
      SUB  BREG          TEST THE LOW PARTS
      SMI                      X
      JMP  DIV6          MAYBE DIVIDE CHECK
DIV1 LDA  AREG          MOVE THE DIVISOR
      STA  ARG3          TO WORKING STORAGE
      LDA  BREG          X
      STA  ARG4          X
      LDA  M30          SET THE COUNTER
      STA  KNTR          FOR 30 BITS
      CRA                      START WITH LOW ORDER QUOTIENT ZERO
DLP  STA  BREG          STORE THE LOW QUOTIENT BITS
      LDA  ARG1          RIGHT SHIFT THE
      LGL  1            DIVIDEND ONE BIT POSITION
      STA  ARG1          X
      LDA  ARG2          X
      LGL  1            X

```

Fixed Point, Double Precision Divide Subroutine (Cont)

	SPL		X
	IRS	ARG1	X
	ANA	MAG	X
	STA	ARG2	X
	LDA	ARG1	FETCH THE HIGH ORDER DIVIDEND
	STA	MPYR	SAVE IN CASE QBIT IS ZERO
	SUB	ARG3	SUBTRACT THE HIGH ORDER DIVISOR
	SPL		TEST FOR POSSIBLE QUOTIENT BIT
	JMP	DIV7	THE QUOTIENT BIT IS ZERO
	STA	ARG1	QUOTIENT BIT MAY BE ONE
	LDA	ARG2	SUBTRACT THE LOW ORDER DIVISOR
	SUB	ARG4	FROM THE LOW ORDER DIVIDEND
	STA	ARG2	AND SAVE THE RESULT
	SMI		TEST FOR A BORROW
	JMP	DIV9	NO BORROW - QUOTIENT BIT IS ZERO
	LDA	ONES	BORROW - SUBTRACT ONE FROM HIGH PART
	ADD	ARG1	X
	SPL		TEST FOR POSITIVE RESULT
	JMP	DIV8	Q BIT IS ZERO - RESTORE DIVIDEND
	STA	ARG1	SAVE THE NEW DIVIDEND
DIV9	IRS	BREG	INSERT THE Q BIT
DIV7	LDA	AREG	LEFT SHIFT THE
	LGL	1	PARTIAL QUOTIENT
	STA	AREG	ONE PLACE
	LDA	BREG	X
	SPL		X
	IRS	AREG	CARRY OVER WORD BOUNDARY
	LGL	1	SHIFT THE LOW Q BITS
	IRS	KNTR	TEST FOR END OF LOOP
	JMP	DLP	NOT YET
	LGR	1	REPOSITION THE
	STA	BREG	LOW Q BITS
	LDA	TEMP	FETCH THE QUOTIENT SIGN
DIVZ	SPL		TEST
	JST	DTCA	THE RESULT IS NEGATIVE
DIVX	JST	ENDX	SETUP THE EXIT CONDITIONS
	IRS	DDIV	STEP TO THE RETURN
	JMP*	DDIV	AND EXIT
	*		
DIV8	LDA	ARG2	RESTORE THE PARTIAL DIVIDEND
	ADD	ARG4	X
	STA	ARG2	X
	LDA	MPYR	X
	STA	ARG1	X
	JMP	DIV7	X
	*		
DIV6	SNZ		NON ZERO IS CERTAIN DIVIDE CHECK
	JMP	DIV5	TEST FOR -1 RESULT
DCK1	JST	DLDA	RESTORE THE ORIGINAL
	DAC	ARG1	DIVIDEND MAGNITUDE
	LDA	SIGN	TEST FOR INITIAL SIGN
	SPL		X
	JST	DTCA	IT WAS NEGATIVE
DCK	JST	DOAS	RECORD THE ERROR
	JMP	DIVX	AND EXIT
	*		
DIV5	LDA	TEMP	TEST THE QUOTIENT SIGN
	SMI		X
	JMP	DCK1	POSITIVE-RESULT IS DIVIDE CHECK
	CRA		FORCE THE QUOTIENT TO -1
	STA	BREG	X
	LDA	M1	X
	STA	AREG	X
	JMP	DIVX	AND EXIT

Output on ASR-33

* THIS SUBROUTINE OUTPUTS ONE CHARACTER TO THE ASR-33. IF THE
* CHARACTER IS PRINTABLE, THE CHARACTER WILL BE PRINTED. IF THE
* ASR-33 PAPER TAPE PUNCH IS ON, THE CHARACTER WILL BE PUNCHED
* (WHETHER PRINTABLE OR NOT). THE SUBROUTINE IS ENTERED WITH THE
* CHARACTER TO BE OUTPUT IN THE A REGISTER.
*

```
REL
SUBR  ASRTYP,STRT      ESTABLISH SUBROUTINE NAME "ASRTYP"
                        HAVING THE ENTRY POINT AT "STRT"
*
*
STRT  DAC    **        SUBROUTINE ENTRY POINT
      SKS    *104      DELAY IF ASR-33 BUSY
      JMP    *-1
      OCP    *104      ENABLE ASR-33 IN OUTPUT MODE
      OTA    4         OUTPUT CHARACTER IN ASCII MODE
      JMP    *-1      (DELAY IF ASR-33 NOT READY)
      JMP*   STRT      RETURN TO CALLING PROGRAM
      END
```

Paper Tape Read Subroutine

* THIS SUBROUTINE READS DATA FROM THE HIGH SPEED PAPER TAPE READER.
* DATA ARE READ TWO CHARACTERS PER ENTRY. THE TWO CHARACTERS ARE PACKED
* INTO A WORD WITH THE FIRST CHARACTER READ IN THE LEFT HALF AND THE
* SECOND CHARACTER READ IN THE RIGHT HALF. THE PACKED WORD IS LEFT IN
* THE A REGISTER ON RETURN TO THE CALLING PROGRAM.
*

```
REL
SUBR  PTR            ESTABLISH SUBROUTINE NAME.
*
PTR   DAC    **      SUBROUTINE ENTRY POINT.
      OCP    1        START TAPE READER.
      INA    *1001    CLEAR A AND INPUT FIRST CHARACTER
      JMP    *-1      (DELAY IF DATA NOT READY).
      LGL    8        POSITION FIRST CHARACTER IN LEFT
*                      HALF OF A.
      INA    1        INPUT SECOND CHARACTER. NOTE THAT
*                      A IS NOT CLEARED AND THAT THE
*                      SECOND CHARACTER IS "ORED" INTO A
      JMP    *-1      (DELAY IF DATA NOT READY).
      OCP    *101     STOP TAPE READER (NOTE THAT TAPE
*                      READER STOPS IN TIME TO PREVENT
*                      LOSING THE FOLLOWING CHARACTER).
      JMP*   PTR      RETURN TO CALLING PROGRAM.
      END
```


Output on High-Speed Paper Tape Punch

* THIS SUBROUTINE PERFORMS THREE DIFFERENT FUNCTIONS
* DEPENDING ON WHICH ENTRY POINT IS USED.
* THE POWER ON ENTRY IS USED TO TURN ON PUNCH
* POWER (THIS IS MADE A SEPARATE FUNCTION BECAUSE
* THE PUNCH WILL NOT BE READY TO RECEIVE DATA UNTIL
* ABOUT FIVE SECONDS AFTER POWER IS TURNED ON
* AND AS A SEPARATE FUNCTION, POWER MAY BE TURNED
* ON PRIOR TO THE PROGRAM BEING READY TO PUNCH
* DATA). THE POWER OFF ENTRY IS USED TO TURN
* PUNCH POWER OFF AFTER THE LAST DATA BLOCK HAS
* BEEN PUNCHED. THE PUNCH DATA ENTRY IS USED TO
* PUNCH A CARD IMAGE (80 CHARACTERS) FROM A PACKED
* DATA FIELD BEGINNING AT THE LOCATION IN A
* DAC FOLLOWING THE SUBROUTINE CALL.

```

      REL
      SUBR  PON          ESTABLISH SUBROUTINE
                          NAME FOR POWER ON
*      SUBR  POFF        ESTABLISH SUBROUTINE
                          NAME FOR POWER OFF
*      SUBR  PNCH        ESTABLISH SUBROUTINE
                          NAME FOR PUNCH DATA
*
*
PON  DAC  **            PUNCH POWER ON ENTRY
      SKS  *102         TEST FOR POWER ON
      OCP  2            IF NOT ON, TURN ON
      JMP* PON          RETURN TO CALLING
                          PROGRAM
*
*
POFF DAC  **            PUNCH PWR OFF ENTRY
      SKS  2            WAIT FOR PUNCH READY
      JMP  *-1          TO ACCEPT DATA TO
                          INSURE AGAINST TURN-
*                          ING PUNCH OFF WHILE
*                          IN PUNCH CYCLE
*
      OCP  *102         TURN PNCH PWR OFF
      JMP* POFF        RETURN TO CALLING
                          PROGRAM
*
*
PNCH DAC  **            PUNCH DATA ENTRY
      LDA* PNCH         SET POINTER TO START
      STA  PTR          OF DATA
      LDA  #-40         SET COUNTER TO -40
      STA  CTR          FOR 40 WORD BLOCK
      SKS  *102         FAIL-SAFE POWER ON
*                          TEST
*      OCP  2            IF POWER OFF TURN ON
*
*
LOOP LDA* PTR          PICK UP PACKED DATA
*                          WORD
      ARR  8            POSITION LEFT CHAR-
*                          ACTER FOR PUNCHING
*      OTA  2            OUTPUT CHARACTER TO
*                          PUNCH
*      JMP  *-1         (DELAY IF PUNCH NOT
*                          READY)
*      ARR  8            REPOSITION RIGHT
*                          CHARACTER FOR PUNCH-
*                          ING
*      OTA  2            OUTPUT CHARACTER TO
*                          PUNCH
*      JMP  *-1         (DELAY IF PUNCH NOT
*                          READY)
*

```

Output on High-Speed Paper Tape Punch (Cont)

	IRS	PTR	STEP DATA POINTER
	IRS	CTR	STEP COUNTER
	JMP	LOOP	IF COUNTER NOT ZERO,
*			CONTINUE PUNCHING
	IRS	PNCH	OTHERWISE, SKIP DATA
*			ADDRESS
	JMP*	PNCH	AND RETURN TO CALL-
*			ING PROGRAM
*			(NOTE THAT PUNCH PO-
*			WER IS NOT TURNED
*			OFF AFTER DATA BLOCK
*			HAS BEEN PUNCHED)
CTR	BSS	1	STORAGE FOR COUNTER
*			VALUE
PTR	BSS	1	STORAGE FOR DATA
*			POINTER
	END		

SECTION VII OPERATION

CONTROL CONSOLE

A separate console unit containing all operating controls and indicators is provided in each DDP-416 system. The console control panel is illustrated in Figure 7-1; the controls and indicators which it contains are described in Table 7-1.

OPERATING PROCEDURES

Turn-On, Turn-Off

To turn on the computer, depress the control panel POWER-ON indicating pushbutton. This operation will apply primary input power to the system and will normalize the status of the machine.

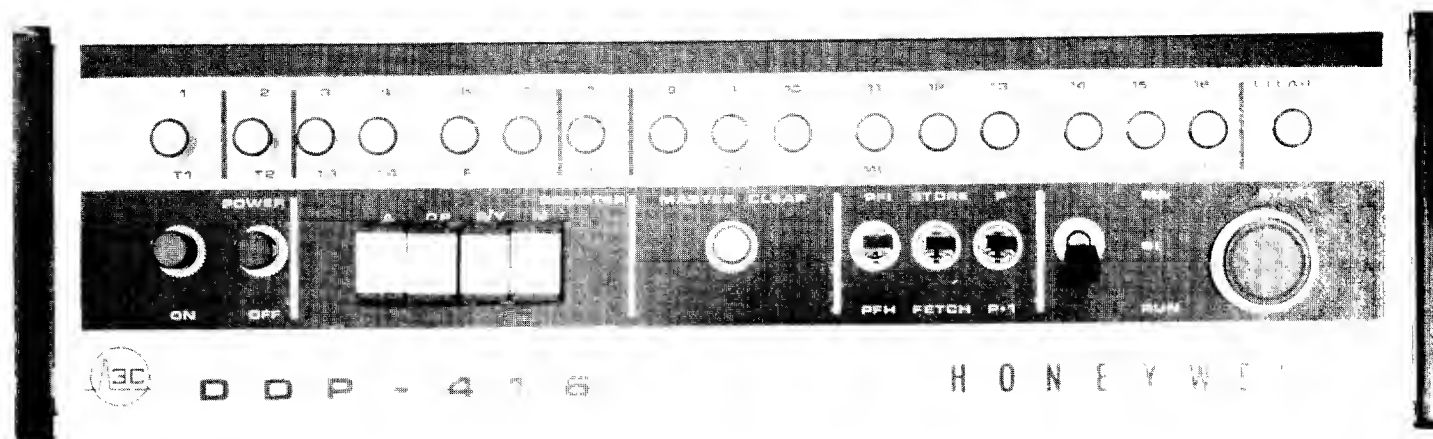
The POWER-ON indicating pushbutton is illuminated when dc power is applied to the memory.

To turn off the computer, depress the control panel POWER-OFF pushbutton. All input power to the system will be removed upon the operation of this control.

Initialize System

When desired, the computer may be initialized by depressing the control panel MSTR CLEAR pushbutton. The operation of this control clears registers A, M, F, P, Y, and SC. All timing levels are set to the state which would exist after the execution of a halt (HLT) instruction (timing level 3 of fetch cycle).

The operation of the MSTR CLEAR control does not affect memory.



A3939

Figure 7-1. Control Console, Control Panel

Table 7-1.
Control Panel Controls and Indicators

Panel Designation	Normal Position	Type	Function
1, 2, ... 15, 16	NA	Indicating Pushbuttons	This control is used for the display of the contents of each bit position of a selected CPU register, for the manual bit-by-bit entry of data bits into the selected register and the display of the states of key control flip-flops.
<p>NOTE</p> <p>These indicators have two sets of panel symbols (upper and lower); those listed in this table (upper set) and those listed in Table 7-2.</p>			
CLEAR	NA	Pushbutton	This control clears any selected (A, P/Y, M) register.
START	NA	Indicating Pushbutton	This control implements the step-by-step reading or insertion of data into consecutive memory locations and for the step-by-step execution of programs. The indicator is illuminated to indicate a RUN condition.
MA/SI/RUN	RUN	Three-Position Toggle Switch	<p>This switch enables the operator to select the operating mode of the computer. The modes selected by each switch position are as follows:</p> <ol style="list-style-type: none"> 1) MA, Memory Access mode during which data may be read and displayed from or inserted into a memory word location. 2) SI, Single Instruction mode which permits the step-by-step execution of a program. 3) RUN, Normal computer operating mode during which the performance of a program may be stopped either by the execution of a halt (HLT) instruction or by the positioning of the mode selector switch to any of the other two positions.

Table 7-1. (Cont)
Control Panel Controls and Indicators

Panel Designation	Normal Position	Type	Function
STORE/FETCH	FETCH	Two-Position Lever-Type Switch	In the FETCH position, this switch permits the readout and display of an addressed memory word location. In the STORE position, this switch enables the insertion of data into an addressed memory word location. If the MA/SI/RUN switch is in SI or RUN mode, the STORE/FETCH switch is disabled.
PFI/PFH	PFH	Two-Position Lever-Type Switch	Used to determine the operation performed by the computer on the detection of power failure. The operation for each position is: <ol style="list-style-type: none"> 1) PFI, the computer will execute a program interrupt when power fails. 2) PFH, the computer will stop when power fails.
P/P+1	P+1	Two-Position Lever-Type Switch	In the P position this switch, in conjunction with other controls and indicators, permits the operator to address and read data from or write data into an addressed memory word location. In the P+1 position, this switch permits the development of consecutive memory location addresses by causing the previous address to be incremented by one. If the MA/SI/RUN switch is in the SI or RUN mode, the P/P+1 switch is disabled.
REGISTER, OP, A, P/Y, M	OP	Interlocked Pushbuttons	The operation of keys A and M cause the contents of the corresponding main frame registers to be displayed at the control panel. <p>The operation of key P/Y causes the contents of the main frame Y register to be displayed at the control panel. If the computer is in an MA mode, the main frame Y and P registers contain the same data; however, in the SI mode the P register contains one more than the Y register. In either case, only the Y register is displayed.</p> <p>The selection of the OP control key causes the states of key control flip-flops to be displayed (refer to Display Operational Data Procedure).</p>

Table 7-1. (Cont)
Control Panel Controls and Indicators

Panel Designation	Normal Position	Type	Function
MSTR CLEAR	NA	Pushbutton	When operated, this control places the computer in a standard cleared state in which registers A, M, P, and Y are cleared; the clock is stopped; and all timing registers are set to the condition which would exist following the execution of a HALT (HLT) instruction. The operation of this control has no effect on memory.
POWER ON	NA	Indicating Pushbutton	The operation of this control applies primary power to the computer. The control is illuminated when dc power for the memory is on.
POWER OFF	NA	Pushbutton	The control, when operated, removes primary power from the computer.

Table 7-2.
Control Panel Data Bit Indicators
Displayed Operational Functions and Indicator Symbols

Applicable Indicator Symbols	Bit Number	Function
T1	1	Timing Level 1
T2	2	Timing Level 2
T3	3	Timing Level 3
T4	4	Timing Level 4
F	5	Fetch Cycle, F
I	6	Indirect Cycle, I
A	7	Execute Cycle, A
PI	9	Permit Interrupt, Indicator illuminated when interrupt permitted
--	10	Unassigned
ML	11	Restricted Mode (Memory Lockout Option)
--	14	Unassigned
MP	15	Memory Parity Error (Memory Parity Option)
P	16	I/O Parity Error (utilized with options which provide parity checking)
<p style="text-align: center;">NOTE</p> <p style="text-align: center;">The operation of the MSTR CLEAR pushbutton will reset all indicators except Timing Level 3 (Bit 3) and Fetch Cycle (Bit 5).</p>		

Read Single Memory Location

The procedure required to access a single memory word location and display the 16-bit contents of the location at the control panel is as follows:

- 1) Set MA/SI/RUN (mode) switch to MA
- 2) Set FETCH/STORE switch to FETCH
- 3) Set P/P+1 switch to P
- 4) Depress P/Y REGISTER control button
- 5) Operate CLEAR pushbutton
- 6) Enter desired memory word location address using data bits indicating pushbuttons (3-16)
- 7) Depress M REGISTER control button

- 8) Operate START pushbutton
- 9) View data bits indicating pushbuttons for desired display

Read Consecutive Memory Locations

To read a series of successive memory word locations without being required to address each location individually, perform the above procedure required to read the first location. Then:

- 10) Set the P/P+1 switch to P+1
- 11) The START pushbutton must be operated each time that the next succeeding location is to be displayed

NOTE

To display at any time (via data bits indicating pushbuttons) the address of the memory location last read from, depress the P/Y REGISTER control pushbutton.

Single Memory Location Data Insertion and/or Modification

The procedure required to access a single memory word location and to enter data into or modify the contents of the location is as follows:

- 1) Set mode switch to MA
- 2) Set FETCH/STORE switch to STORE
- 3) Set P/P+1 switch to P
- 4) Depress P/Y REGISTER control button
- 5) Operate CLEAR pushbutton
- 6) Enter desired memory word location address using data bits indicating pushbuttons (3-16)
- 7) Depress M REGISTER control button
- 8) Enter desired word or modification bits using data bits indicating pushbuttons
- 9) Operate START pushbutton to complete operation

Consecutive Memory Location Data Insertion And/Or Modification

To insert the same word into a series of successive memory locations without being required to address each location individually, perform the above procedure required for the first word, then:

- 10) Set the P/P+1 switch to P+1
- 11) The START pushbutton must be operated to initiate each successive insertion of data

NOTE

To display at any time (via data bits indicating pushbuttons) the address of the memory location last written into, depress the P/Y REGISTER control button.

Single-Step Program Execution

A stored program may be examined in detail by executing it step-by-step in the following manner:

- 1) Set mode switch to SI
- 2) Operate MSTR CLEAR pushbutton
- 3) Select desired register using REGISTER control buttons A or P/Y
- 4) Enter desired initial values using data bits indicating pushbuttons
- 5) Operate START pushbutton. This causes the first instruction to be fetched and loaded into the M register. The display of the first instruction is obtained by the operation of REGISTER control button M. After the first instruction, each time the START pushbutton is operated the previously fetched instruction is executed, the next instruction is fetched, the P register is incremented, and the computer stops. The operation of REGISTER control button P/Y at this time will cause the data bits indicators to display the Y register which will contain the address from which the new instruction was fetched. Note that the P register, which is not displayed, will contain a value one higher than that in the Y register.

Read Mainframe Register

To display the contents of mainframe registers A, Y, or M at the control panel, depress the corresponding REGISTER button. Bits 1 through 16 of the selected register will be displayed by the control panel data bits (1 through 16) indicating pushbuttons. An illuminated indicator designates a ONE bit, an unlighted indicator a ZERO bit.

Display Operational Data

The operation of the OP REGISTER control button causes the operational status of the circuits within the computer to be displayed by the control panel data bits indicators. The functional status indicated by each of the 16 indicating pushbuttons and the symbol applicable to each when displaying operation data are described in Table 7-2.

Run Program

To enable the computer to perform a program in its normal RUN mode, do the following:

- 1) Set mode switch to RUN
- 2) Operate MSTR CLEAR pushbutton
- 3) Depress P/Y REGISTER control button
- 4) Enter Starting address using the data bits indicating pushbuttons 1 - 16.

NOTE

If this is a program restart procedure, it may be necessary to modify or enter new data into other registers of the computer at this time. The procedure would be to operate the appropriate REGISTER control button and enter the necessary data using the data bits controls.

- 5) Operate START pushbutton to initiate automatic execution of program. The selected program will run until either a HLT (halt) instruction is performed or the mode switch is set to the SI position. In either case, the computer is stopped in a standard operating condition and can be restarted in either the RUN or SI mode.

APPENDIX A NUMBERING SYSTEM AND TWO'S COMPLEMENT ARITHMETIC

Binary Numbering System

Sixteen-bit data words are stored in two's complement notation. The most significant bit of a data word may be considered to be the arithmetic sign of the number represented. The MSB is ZERO for positive (+) numbers and a ONE for negative (-) numbers. Bits 2 through 16 of the data word represent the value in binary form. Positive values thus range from zero (which always has a positive sign) to 32,767 as follows:

0	000 000 000 000 000	Zero
0	000 000 000 000 001	+ 1
0	000 000 000 000 010	+ 2
	'	'
	'	'
	'	'
	'	'
0	111 111 111 111 111	+ 32,767

Negative numbers are represented in two's complement form and always have a ONE in the sign bit position.

Two's Complement Arithmetic

The two's complement of a binary number is obtained by complementing (reversing) each bit and adding one. For example, the two's complement of +1, which represents -1, is obtained as follows:

+ 1	0	000 000 000	000 001
Complement	1	111 111 111	111 110
Add 1	0	<u>000 000 000</u>	<u>000 001</u>
Two's Complement (-1)	1	111 111 111	111 111

The number range for negative values is from -1 to -32,768 as follows:

1	111 111 111 111	111	(-1)
1	111 111 111 111	110	(-2)
1	111 111 111 111	101	(-3)
1	000 000 000 000	000	(-32,768)

If +1 is added to -1, the result is zero. Thus:

1	111 111 111 111	111	-1
0	000 000 000 000	001	+1
<hr/>			
0	000 000 000 000	000	Zero

Note that a carry bit from the most significant position has been ignored. In two's complement arithmetic, if numbers of unlike signs are added together, carries from the most significant bit are disregarded.

Overflow

Overflow is the condition that occurs when two numbers of like signs are added together to produce a sum of a different sign. For example, adding +1 to $\pm 32,767$ would produce a result larger than the capacity of a single data word.

0	111 111 111 111	111	(+32,767)
0	000 000 000 000	001	(+1)
<hr/>			
1	000 000 000 000	000	

The different sign of the result defines an overflow condition.

Addition on the computer is performed by adding a quantity in the memory to a quantity in the A-register. True signed arithmetic takes place.

APPENDIX B ASCII CODE*

Standard Code

<div><div><div>b7b6b5</div><div>b4b3b2b1</div><div>Bits</div></div><div><div>Column</div><div>Row</div></div></div>					0	1	2	3	4	5	6	7	
	0	0	0	0	0	NUL	DLE	SP	0	`	P	@	p
	0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
	0	0	1	0	2	STX	DC2	"	2	B	R	b	r
	0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
	0	1	0	0	4	EOT	DC4	\$	4	D	T	d	t
	0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
	0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
	0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
	1	0	0	0	8	BS	CAN	(8	H	X	h	x
	1	0	0	1	9	HT	EM)	9	I	Y	i	y
	1	0	1	0	10	LF	SS	*	:	J	Z	j	z
	1	0	1	1	11	VT	ESC	+	;	K	[k	{
	1	1	0	0	12	FF	FS	,	<	L	~	l	
	1	1	0	1	13	CR	GS	-	=	M	J	m	}
	1	1	1	0	14	SO	RS	.	>	N	^	n	
	1	1	1	1	15	SI	US	/	?	O	_	o	DEL

Character Representation

The standard 7-bit character representation, with b₇ the high-order bit and b₁ the low-order bit, is shown below.

Example. The bit representation for the character "K", positioned in column 4, row 11, is:

b₇ b₆ b₅ b₄ b₃ b₂ b₁
1 0 0 1 0 1 1

The code table position for the character "K" may also be represented by the notation "column 4, row 11" or alternately as "4/11." The decimal equivalent of the binary number formed by bits b₇, b₆ and b₅, collectively, forms the column number, and decimal equivalent of the binary number formed by bits b₄, b₃, b₂ and b₁, collectively, forms the row number.

Legend

Control Characters

NUL	Null	DC3	Device Control 3
SOH	Start of Heading (CC)	DC4	Device Control 4
STX	Start of Text (CC)		(stop)
ETX	End of Text (CC)	NAK	Negative Acknowledge (CC)
EOT	End of Transmission (CC)	SYN	Synchronous Idle (CC)
ENQ	Enquiry (CC)	ETB	End of Transmission Block (CC)
ACK	Acknowledge (CC)	CAN	Cancel
BEL	Bell (audible or attention signal)	EM	End of Medium
BS	Backspace (FE)	SS	Start of Special Sequence
HT	Horizontal Tabulation (punched card skip) (FE)	ESC	Escape
LF	Line Feed (FE)	FS	File Separator (IS)
VT	Vertical Tabulation (FE)	GS	Group Separator (IS)
FF	Form Feed (FE)	RS	Record Separator (IS)
CR	Carriage Return (FE)	US	Unit Separator (IS)
SO	Shift Out	DEL	Delete
SI	Shift In		
DLE	Data Link Escape (CC)		
DC1	Device Control 1		
DC2	Device Control 2		

(CC) Communication Control. (FE) Format Effector. (IS) Information Separator.

Graphic Characters

Column/Row	Symbol	Name	Column/Row	Symbol	Name
2/0	SP	Space (normally nonprinting)	2/15	/	Slant
2/1	!	Exclamation Point	3/10	:	Colon
2/2	"	Quotation Marks (diaeresis)	3/11	;	Semicolon
2/3	#	Number Sign	3/12	<	Less Than
2/4	\$	Dollar Sign	3/13	=	Equals
2/5	%	Percent	3/14	>	Greater Than
2/6	&	Ampersand	3/15	?	Question Mark
2/7	'	Apostrophe (closing single quotation mark; acute accent)	4/0	`	Grave Accent (opening single quotation mark)
			5/11	[Opening bracket
			5/12	~	Tilde
			5/13]	Closing bracket
2/8	(Opening Parenthesis	5/14	^	Circumflex
2/9)	Closing Parenthesis	5/15	_	Underline
2/10	*	Asterisk	6/0	@	Commercial at
2/11	+	Plus	7/11	{	Opening brace
2/12	,	Comma (cedilla)	7/12	~	Overline
2/13	-	Hyphen (minus)	7/13	}	Closing brace
2/14	.	Period (decimal point)	7/14		Vertical line

*The information presented is an excerpt from the proposed revised American Standard Code for Information Interchange.

APPENDIX C
SUMMARY OF STANDARD INSTRUCTIONS
(Listed in Alphabetical Order)

<u>Mnemonic</u>	<u>Octal Code</u>	<u>Instruction</u>	<u>Type</u>	<u>Execution Time (μsec)</u>
ADD	06	Add	MR	1.92
ALR	0416	Logical Left Rotate	SH	$0.96 + 0.48n$
ALS	0415	Arithmetic Left Shift	SH	$0.96 + 0.48n$
ANA	03	AND to A	MR	1.92
ARR	0406	Logical Right Rotate	SH	$0.96 + 0.48n$
ARS	0405	Arithmetic Right Shift	SH	$0.96 + 0.48n$
CRA	140040	Clear A	G	0.96
ENB	000401	Enable Program Interrupt	G	0.96
ERA	05	Exclusive OR to A	MR	1.92
HLT	000000	Halt	G	
INA	54	Input to A	IO	1.92
INH	001001	Inhibit Program Interrupt	G	0.96
IRS	12	Increment, Replace and Skip	MR	2.88
JMP	01	Unconditional Jump	MR	0.96
JST	10	Jump and Store Location	MR	2.88
LDA	02	Load A	MR	1.92
LGL	0414	Logical Left Shift	SH	$0.96 + 0.48n$
LGR	0404	Logical Right Shift	SH	$0.96 + 0.48n$
NOP	101000	No Operation	G	0.96
OCF	14	Output Control Pulse	IO	1.92
OTA	74	Output From A	IO	1.92
SKP	100000	Unconditional Skip	G	0.96
SKS	34	Skip if Ready Line Set	IO	1.92
SMI	101400	Skip if A Minus	G	0.96
SMK	74	Set Mask	IO	1.92
SNZ	101040	Skip if A Not ZERO	G	0.96
SPL	100400	Skip if A Plus	G	0.96
STA	04	Store A	MR	1.92
SUB	07	Subtract	MR	1.92
SZE	100040	Skip if A ZERO	G	0.96

APPENDIX D PERIPHERAL DEVICE COMMANDS

ASR 33/35 Model 416-53/55

OCP	0004	Enable ASR-33/35 In Input Mode
OCP	0104	Enable ASR-33/35 In Output Mode
SKS	0004	Skip if ASR-33/35 is Ready
SKS	0104	Skip if ASR-33/35 is Not Busy
SKS	0204	Skip if ASR-33/35 is Ready
SKS	0404	Skip if ASR-33/35 is Not Interrupting
SKS	0504	Skip if Stop Code Was Not Read on ASR-33/35
INA	0004	Input in ASCII from ASR-33/35
INA	0204	Input in Binary from ASR-33/35
INA	1004	Clear Register A and Input in ASCII from ASR-33/35
INA	1204	Clear Register A and Input in Binary from ASR-33/35
OTA	0004	Output in ASCII to ASR-33/35
OTA	0204	Output in Binary to ASR-33/35
SMK	0020	Set Interrupt Mask (A_{11})

High Speed Paper Tape Reader - Model 416-50

OCP	0001	Start Paper Tape Reader In Forward Direction
OCP	0101	Stop Paper Tape Reader
SKS	0001	Skip if Paper Tape Reader is Ready
SKS	0401	Skip if Paper Tape Reader is Not Interrupting
INA	0001	Input from Paper Tape Reader
INA	1001	Clear Register A and Input From Paper Tape Reader
SMK	0020	Set Interrupt Mask (A_9)

High Speed Paper Tape Punch - Model 416-52

OCP	0002	Enable Paper Tape Punch
OCP	0102	Turn Paper Tape Punch Power Off
SKS	0002	Skip if Paper Tape Punch is Ready
SKS	0102	Skip if Paper Tape Punch is Enabled
SKS	0402	Skip if Paper Tape Punch is Not Interrupting
OTA	0002	Output To Paper Tape Punch
SMK	0020	Set Interrupt Mask (A_{10})

APPENDIX E
MAIN FRAME OPTION COMMANDS

<u>Mnemonic</u>	<u>Octal Code</u>	<u>Instruction</u>	<u>Type</u>	<u>Execution Time</u>
<u>Memory Parity - Model 416-07</u>				
RMP	000021	Reset Memory Parity Error	G	0.96
SMK '0020	170020	Set Interrupt Mask (A ₁₅)	IO	1.92
SPN	100200	Skip if No Memory Parity Error	G	0.96
SPS	101200	Skip if Memory Parity Error	G	0.96
<u>Memory Lockout - Model 416-08/-08-1</u>				
ERM	001401	Enter Restricted Mode	G	0.96
SMK '1320	171320	Set Relocation Register	IO	1.92
SMK '1420	171420	Set Lockout Mask 1	IO	1.92
SMK '1520	171520	Set Lockout Mask 2	IO	1.92
<u>Direct Memory Access (DMA) - Model 416-21</u>				
INA '1124	171124	Read Range Counter Channel 1	IO	1.92
INA '1224	171224	Read Range Counter Channel 2	IO	1.92
INA '1324	171324	Read Range Counter Channel 3	IO	1.92
INA '1424	171424	Read Range Counter Channel 4	IO	1.92
SMK '0124	170124	Load Address Counter Channel 1	IO	1.92
SMK '0224	170224	Load Address Counter Channel 2	IO	1.92
SMK '0324	170324	Load Address Counter Channel 3	IO	1.92
SMK '0424	170424	Load Address Counter Channel 4	IO	1.92
SMK '1124	171124	Load Range Counter Channel 1	IO	1.92
SMK '1224	171224	Load Range Counter Channel 2	IO	1.92
SMK '1324	171324	Load Range Counter Channel 3	IO	1.92
SMK '1424	171424	Load Range Counter Channel 4	IO	1.92
<u>Priority Interrupt - Model 416-25</u>				
SMK '0120	170120	Set Interrupt Mask Lines 1-16	IO	1.92
SMK '0220	170220	Set Interrupt Mask Lines 17-32	IO	1.92
SMK '0320	170320	Set Interrupt Mask Lines 33-48	IO	1.92

APPENDIX E (Cont)
MAIN FRAME OPTION COMMANDS

<u>Mnemonic</u>	<u>Octal Code</u>	<u>Instruction</u>	<u>Type</u>	<u>Execution Time</u>
<u>Real Time Clock - Model 416-12</u>				
OCP '0220	030220	Reset Interrupt Request and Stop Clock	IO	1.92
OCP '0020	030020	Reset Interrupt Request and Run Clock	IO	1.92
SKS '0020	070020	Skip if RTC not interrupting	IO	1.92
SMK '0020	070020	Set Interrupt Mask (A ₁₆)	IO	1.92

APPENDIX F DEDICATED LOCATIONS

<u>Octal Address</u>	<u>Assignment</u>
00001 } thru } 00017 }	Protected Fill Program
00020 00021	Starting } Addresses for Final } DMC Channel 1
00022 } thru } 00057 }	DMC Channels 2 thru 16
00060	Power Failure Interrupt Link
00061	Real Time Clock
00062	Lockout Violation Interrupt Link
00063	Standard Interrupt Link
00064	Optional PI No. 1 Link
00065 } thru } 00143 }	Optional PI No. 2 thru 48 Links